FAKULTÄT FÜR !INFORMATIK

Visualisation and Verification of Ontology Alignment Results

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur/in

im Rahmen des Studiums

Intelligente Systeme

eingereicht von

Slaven Banovic
Matrikelnummer 9625801

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung:
Betreuerin: Dipl.-Ing. Dr.techn. Monika Lanzenberger

Wien, 30. Oktober 2009

(Unterschrift Verfasser/in)

(Unterschrift Betreuer/in)

Wien, 30. Oktober 2009

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit — einschließlich Tabellen, Karten und Abbildungen —, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Slaven Banovic

Slaven Banovic

Visualisation and Verification of Ontology Alignment Results

dedicated to my daemons and muses
real or imagined

Abstract

Ontologies play an important role in the Semantic Web in providing an annotated knowledge base and promoting interoperability of applications. The alignment of different ontologies remains to be a major problem in this research field, which has concentrated on the development of automated mapping tools. Whereas the mapping algorithms show promising results it is evident, that a human supervisor is still necessary to attend the process.
In this thesis we implement a novel approach to ontology alignment by adopting techniques from information visualisation (InfoVis) to support the cognitive process and alleviate the human work load. We discuss the task description of visual ontology alignment and identify crucial requirements for the successful implementation of the methodology.
A vital subtask of the methodology of visual ontology alignment is the verification of ontology alignment results. We present a prototype tool that implements a multi-view visualisation to enable ontology verification and discuss the development of future versions.
The ultimate goal of AlViz is to fully implement the task description of visual ontology alignment. To evaluate our research a comprehensive evaluation strategy is outlined.

Abstrakt

Ontologien sind die Grundpfeiler des Semantic Web. Sie dienen als Wissensdatenbank und ermöglichen die Interaktion zwischen unterschiedlichen Applikationen. Ontology Alignment ist weiterhin eines der größten Probleme dieses wissenschaftlichen Forschungsfeldes. Bestehende Ansätze konzentrieren sich zur Zeit vorwiegend auf die Entwicklung und Optimierung von automatisierten Mapping Tools. Während immer bessere Resultate erziehlt werden, ist eine begleitende Steuerung durch einen Menschen unerlässlich.
In dieser Arbeit implementieren wir einen neuartige Verfahrensweise, die Anleihen aus Informationsvisualisierung (InfoVis) nimmt. Visuelle Artefakte unterstützen den kognitiven Denkprozess und erleichtern komplexe Informationsverarbeitungen. Wir beschreiben ausführlich die Konzepte hinter Visual Ontology Alignment und identifizieren Parameter, die für eine erfolgreiche Implementierung des Prozesses, notwendig sind.
Ein wichtiger Teilschritt in diesem Unterfangen stellt die Verifikation von Ontology Alignment Ergebnissen dar. Hierfür stellen wir einen Prototypen einer Applikation vor, der diesen Teilbereich durch die Darstellung einer Multi-View Umgebung meistert.
Ziel von zukünftigen Versionen von AlViz ist es, ein Werkzeug zur Verfügung zu stellen, dass den gesammten Bereich von Visual Ontology Alignment abzudecken weiss. Abschliessend präsentieren wir eine Evaluierungsstrategie um die gewonnenen Erkenntnisse verifizieren zu können.

Contents

List of Figures

List of Tables

List of Listings

Chapter 1

Introduction

1.1 Motivation

The World Wide Web is considered to be the most excessive data repository available. In July 2008 Google[1] announced [Alpert and Hajaj, 2008] that their system that processes links on the web has reached a total of 1,000,000,000,000 unique URLs. This is, of course, just a rough estimate, considering that minimal effort has been taken to filter similar page content. However, the number of individual web pages is growing by several billion pages per day. Additionally, "[m]ost of the Web's information is buried far down on dynamically generated sites, and standard search engines never find it." [Bergman, 2001] Even though significant progress has been made [Madhavan et al., 2008] to recover information from the so called "Deep Web", most of the Web's content remains unseen.

Most of these web pages have been created to be viewed by a human being, using a web browser to render text content, pictures, and multimedia objects. The latter poses quite a difficulty for automated search engines to gather reliable information. Traditional web crawling algorithms are limited in their perception of a web site to text parsing and usually ignore multimedia content. In a simplified view of a web spider, the basic concept is to parse the plain text of a web site and create an index of words [Brin and Page, 1998]. A query request on the resulting database is answered by a simple pattern matching approach, yielding query results that match the search term. The key technology behind any web search engine is its ranking mechanism, which is usually kept as a secret. A naïve ranking approach includes the number of occurrences of a specific word on the original web site, or if the term has been marked with a specific HTML tag. Of course there are more elaborated ranking

[1] http://google.com, (30.10.2009)

systems but they all have in common that they are only operating on a syntactic level. They try to identify a text token by syntactic means.

The inability of search engines to act on a semantic level is obvious: There are few semantic annotations which can be identified on a syntactic level. The syntactic structure provided by HTML tags do not imply semantic connotation. While Computational Linguistics made some progress in automated processing of natural language in the last decade, results are closely dependent on the domain of the analysed text. Basically a properly identified token is matched against a predefined index, containing the intended meaning of the corresponding word [Krovetz and Croft, 1992]. Therefore the accuracy of the results is highly dependent on the given database of mappings between terms and appropriate meaning.

The evolution of the Web from a linked data repository to a distributed knowledge base has been deployed with the concept of the Semantic Web [Berners-Lee et al., 2001]. The Semantic Web is built on top [Hendler, 2001] of the World Wide Web and offers an extension of formal language used to describe the content of the appropriate website in a machine readable form. It provides additional semantic annotations to syntactic structures, already defined within the World Wide Web. The goal of the Semantic Web is to provide a basic structure to exploit the vast range of available information and enable both human and machine agents to search the knowledge base more effectively. More precisely the Semantic Web intends to augment semantic meaning to the information resources within and provides this information to human and machine agents.

Ontologies are a common technique to provide semantic content to a domain and are seen as key method for the Semantic Web. The most cited definition of an ontology has been presented by Gruber [1993]: “An ontology is an explicit specification of a conceptualization”. Since then, the definition has been extended by three additional conditions that are summarised by Sampson [2007]: “An ontology is an explicit, formal specification of a conceptualization of a domain of interest”. Technically an ontology is an hierarchical taxonomic scheme rendering the categories and classes of a domain. These tokens may be enhanced by additional properties which facilitate reasoning and inference support. A thoroughly crafted ontology is a knowledge base, comprised of a set of explicit formal specifications of the terms in the domain.

> An ontology defines a common vocabulary for researchers who need to share information in a domain. It includes machine-interpretable definitions of basic concepts in the domain and relations among them. [Noy and McGuinness, 2001]

The compilation of an ontology of a domain is usually done for a very specific task or application by a domain expert. This procedure requires explicit knowledge of both the appropriate domain and the field of application the ontology is used for.

The granularity of the ontology is dependent on the required task. There is no need to expand the complexity of the ontology if the additional classes are not used by the underlying application.

The Semantic Web is supposed to inherit the universality and the expressive power of the World Wide Web, which comes at a price: The Semantic Web will — just as the World Wide Web — be a decentralised collection of interconnecting, yet heterogeneous content. For a application to successfully traverse and navigate reasonably within the Web, it has to be equipped with a structured collection of information and a set of inference rules (i.e. an ontology). However, based on the heterogeneous nature of the Web, machines and applications will be confronted with resources based on deviant or conflicting ontologies. And yet, for the Semantic Web to emerge, these issues have to be resolved.

To enable interoperability between different applications the corresponding ontologies need to be aligned. Due to their individual process of creation, the setup and the naming conventions used may differ between the ontologies. Additionally, even equally named classes may hold different semantic meanings. There is no way to avoid ambiguity of natural language. In fact, ontology alignment is a non-trivial task which requires human interaction. Due to the underlying semantic content current automated approaches yield no satisfactory results. However, modern automated approaches provide results that can be used as an intermediate alignment for a human ontology engineer to complete the alignment process manually. To this end the computer generated list of preliminary alignments has to be refined and prepared to be human readable.

The concept presented by Lanzenberger and Sampson [2006] introduces visualisation as a promising tool for ontology alignment, called AlViz. The technical approach is built upon the human capabilities to conceive visual objects far better than naked numbers and strings. The two compared ontologies are transcoded and rendered as two distinct three-dimensional graphs, enabling the viewer to capture the domain in multiple facets. While placing the graphs of the ontologies next to each other, matching nodes are coloured according to the given alignment results. Additionally, on the left-hand side each ontology is rendered in a tree view, showing its hierarchical structure. Both graph and tree view are linked with each other, highlighting the appropriate nodes when selected. Moreover, the ontology visualisations are linked with each other, enabling the viewer to recognize matching nodes.

The described layout empowers the ontology engineer to quickly comprise the ontologies, overview the intersections, and register the similarities. Additionally, possible miscalculations of the automated alignment process may be recognised by browsing the neighbourhood of the appropriate nodes both in the graph and tree views. This supplemental process is supposed to introduce a significant improvement to ontology alignment.

1.2 Objectives

The major objectives of this thesis are twofold: First, the identification of necessary requirements for the novel approach of visual ontology alignment. For this cause, we discuss the specifications of ontology authoring, compare representation alternatives, and analyse competing requirements for visualising ontologies and ontology alignments. These findings aid us in the process of identification of suitable visualisation methods that render an intuitive representation of ontology alignments. Ultimately, we need a comprehensive compilation of requirements for visual ontology alignment that composes from the results in the aforementioned survey.

The second objective is to implement a prototype application that incorporates the requirements and proves its reliability for the task of ontology alignment verification. We aim to combine information visualisation techniques with the process of ontology alignment and develop an interactive visualisation for ontology alignment results. The requirements from the first part of this thesis aid us in the development process.

The objectives of this thesis have to be placed in the context of the recent visual ontology alignment research and the conceptualisation and development of AlViz. Lanzenberger and Sampson [2006] outline the initial conceptualisation and propose the use of visualisation techniques to facilitate user understanding of ontology alignment results. Sampson and Lanzenberger [2006] propose the research question, how visualisation can support the ontology engineer's understanding and interpretation of alignment results, and introduce visual ontology alignment as a novel approach. Sampson et al. [2007] describe an integrated approach to organizational data interoperability with the help of visual ontology alignment. Lanzenberger and Sampson [2007] extend a general alignment framework to reflect the adoption of visualisation techniques and evaluate visual ontology alignment in multiple stages. Sampson [2007] describes in her Ph.D. thesis a comprehensive framework for understanding ontology alignment quality, including ontology alignment definition and description, conceptual framework development, ontology quality and evaluation measures, data model quality, and ontology management. The interoperability of Semantic Web applications are discussed by Sampson et al. [2008]. The paper proposes the use of ontologies and conceptual modelling in an overall approach for data interoperability and presents promising results from a tool evaluation for locating additional candidate alignments. Lanzenberger et al. [2008] discuss quality measures for ontology alignment and define requirements for visual ontology alignment.

Additionally, two master theses depict technical enhancements and essential development achievements of the implementation of AlViz. Huber [2009] improves the clustering algorithm to achieve a balanced visualisation, and conducts an evaluation study to validate the improvement in usability. Gradwohl [2009] pursues the implementation of a suitable work flow for ontology alignment visualisation

by implementing re-calculation of alignment results. Furthermore, he extended the Euzenat et al. [2004] alignment format to incorporate additional requirements used by AlViz.

1.3 Thesis Outline

The embracing research field of this thesis is the Semantic Web. We start with the history and definition of the next generation of the World Wide Web in Chapter 2 and continue to lay out the foundations of ontology authoring by discussing the applications and development of ontologies. An in-depth analysis of the problems of ontology mapping and ontology alignment, and the presentation of a standardised XML format for the representation of ontology alignment results completes the necessary background knowledge of ontology authoring.

We further engage with an introduction of information visualisation techniques for the representation of ontologies and ontology alignments. An analysis of requirements for alignment visualisation and promising graph visualisation methods concludes the chapter.

In Chapter 3 we discuss the implementation of the novel approach of visual ontology alignment. We outline the task description and introduce requirements for a successful implementation. We introduce a prototype application for ontology validation using a multi-view visualisation. AlViz combines automated mapping algorithms and interactive visualisation techniques to provide a multiple representations of alignment augmented ontologies.

In Chapter 4 we describe a use case scenario and outline the necessary steps to conclude a validation process of prepared alignment mappings using the visualisation capabilities of AlViz. We prepare a conceptual evaluation strategy to verify user performance and usability of the methodology. Thereafter we give a detailed description of future enhancements to achieve the goal of providing a tool for visual ontology alignment.

Finally, in Chapter 5 we compile a summary and present our conclusions. The chapter ends with an outline of future work.

Chapter 2

Foundations

2.1 Semantic Web

The underlying structure of today's World Wide Web is based on a simple principle. Web content consists mainly of distributed hypertext which is unidirectionally connected. Access is enabled by a combination of search by pattern matching and link navigation. The content is primarily designed to be accessed and interpreted by humans. The underlying hypertext paradigm is mainly concerned with layout and presentation issues. The meaning of the content is implicit and subject to the reader's interpretation. Additionally, the increasing use of images and multimedia content introduces multi-modal dependencies in human–computer interaction. Whereas static textual context is supposed to be perceived visually, the amendment of animated images, sound, and interactive content requires multiple sensory receptors for the user (or machine agent) to perceive and comprehend the provided information. This tendency of increased modality penalises or even obviates users with cognitive or sensory impairments. If the information is provided by a combination of multiple modalities, the content is only comprehensible when all communication paths are received. Similar difficulties arise for automated processing or software agents.

Simplicity has been one of the key factors for the fast growth of the Web, enabling untrained users to navigate within and even create new content. The drawback is a lack of expressiveness of meaning. The HyperText Markup Language[1] (HTML) has been developed to provide easy annotations for structural tagging. The content can be tagged on a syntactic level (e.g. title, subtitle, paragraph, emphasized text, ...) but not by semantic meaning. This shortcoming has been targeted by the Semantic Web.

[1] http://www.w3.org/TR/html4/, (30.10.2009)

The key idea behind the Semantic Web is to explicate the meaning of web content by adding semantic annotations to syntactic structures, and to provide reasoning methods for inference. "The goal [...] is to transform the Web from a linked document repository into a distributed knowledge base and application plattform [Horrocks, 2007]."

The transformation implies a shift of interpretation. The meaning of the content becomes explicitly defined and linked to a well defined and publicly available vocabulary. The exploitation of the vocabulary enables automated processes and software agents to *understand* the content and navigate the web more efficiently. They can even assist human users with cognitive or sensory impairments.

> The semantic web is an extension of the World Wide Web in which both data and its semantic definition can be processed by computer programs. The next generation of the Web will combine existing web technologies with knowledge representation formalisms in order to provide an infrastructure allowing data to be processed, discovered and filtered more effectively in the web [Grau, 2004].

As we mentioned previously, the concept of the Semantic Web is based on extending the basic structure and appending semantic annotations to already existing syntactical structures. Foremost the process of extension is a simple tagging mechanism. The eligible term gets classified and annotated with the appropriate class. Without an underlying knowledge base this tagging would be useless as no new informations has been obtained. The problem of understanding the term has only been transformed to a problem of understanding the class name. So far no meaning has been introduced.

In order to solve the latter we need to access a knowledge base providing sufficient semantic information about the class we have identified. The eligible term becomes an instance of the class in an appropriate knowledge base rendering a semantic model of a domain. This model is being provided by ontologies.

2.2 Ontology

Historically, the term *ontology* has been borrowed from philosophy where its roots can be traced back to ancient Greek philosophy. In computer science ontologies are a rather new concept, but, similar to the philosophical counterpart, had and still has an ongoing discussion about its definition. However, the discussion in computer science will not withstand the quality of discussion in philosophy, as in the former, the dispute covers mostly technical and syntactical differences. Øhrstrøm et al. [2005] presents a comprehensive overview and comparison.

In artificial intelligence literature you can find many different and sometimes even contradicting definitions. In the scope of this work an ontology represents a model of a domain. It presents an ”explicit, formal specification of a conceptualisation of a domain of interest.” [Sampson, 2007] All entities of the ontology are explicitly defined and formally specified, thus alleviating automated parsing and machine processing. Ontologies are supposed to render a (task-)complete model of a limited domain of interest. They are not meant to model the whole world, but to provide sufficient information of a particular domain of interest to fulfil the requirements of a given task. Depending on the level of detail, ontologies can be used as a controlled vocabulary, glossary, thesaurus, or all of the above. They can deliver a list of unambiguous interpretations of terms, provide additional semantic meanings to these terms, and yield the relationships that hold between them. Ontologies are tailored to act as a knowledge base on a certain subject.

2.2.1 Ontology Definition

An ontology consists of a vocabulary describing the aspects of the domain, provides properties and semantic specifications to every specified item and aligns them in a natural order by referencing individual items based on the relationships between them. Every entity of the ontology holds a semantic definition of a single part of the model, giving an explicit specification of the intended meanings to the terms in the vocabulary. So in the context of an ontology, every entity has a formally specified definition that yields a semantic meaning of this entity.

Formally an ontology consists of classes (explicit description of concepts in a domain), properties (features, attributes and restrictions of a class, relations that hold between individual classes) and instances (i.e. a real world object of a class of the model). Similar to Ehrig [2005] we define an ontology O as a tuple:

$$O := (C, \leq_C, R, \leq_R, I, \leq_I, A)$$

- with disjoint sets C (classes), R (relations and properties) and I (instances),
- partial orders $\leq_C$ on C, $\leq_R$ on R and $\leq_I$ on I holding the hierarchy of classes, relations and instances,
- and A representing the axioms used for inferring knowledge.

The classes are arranged in a taxonomic hierarchy, where the root of the tree is usually defined as an abstract term (e.g. *Thing* in OWL, see Section 2.2.4 for details), whereof every other entity can be derived. The taxonomic arrangement emerges naturally from the relations between the classes. Every class is a subclass of its predecessor defined by a *is-a* relation, thus forming an hierarchical structure. A subclass represents concepts that are more specific than its superclass. Additional prop-

erties and constraints can be specified to explicitly define relations among classes. Multiple subclasses of a single superclass (i.e. sibling classes) must be defined properly to yield a semantic disambiguation. This is achieved by appropriate restrictions enforcing to model semantic differences of adjacent classes described by their properties.

The structure defined by classes and properties lays out a framework which is usually referred to as *core ontology schema*. The schema defines the relations between concepts and properties and forms an explicit rule set which can be used to infer knowledge. A core ontology schema together with a set of individual instances of classes and a set of axioms constitutes an ontology[2].

Another widely accepted definition according to Noy and McGuinness [2001] states, that an ontology consists of classes, properties and restrictions. Thus, an ontology together with a set of individual instances constitutes a knowledge base. However, there is a fine line where the ontology ends and the knowledge base begins, depending on its actual usage. The level of granularity can differ among different applications, such that the leaf nodes of an ontology can be interpreted both, as classes or as instances of the appropriate ontology.

In order to understand the conception of ontologies, a visual example will be helpful. The following is a graph representation of the science ontology[3], which is a slightly improved version of the KA[2] ontology developed by the Knowledge Annotation Initiative of the Knowledge Acquisition Community[4]. Notice that these are just examples to introduce a common understanding of ontologies. Neither the presented ontology nor the visualisation tools represent a standardised view of the subject. An ontology is a rather abstract conception. The actual implementation depends on the use case, environmental constraints, or personal preferences. The visualisation of an ontology describes a graphical interpretation of a conceptualisation of a domain: They do not claim to be correct but rather useful for a given task.

Figure 2.1 shows the class hierarchy of the science ontology. It represents a graphical representation of the class taxonomy with the root node being *Thing*. Note that the ontology hierarchy is a partial order: Every subclass inherits the properties of its predecessor, thus incorporating a specialised entity of its parent.

The screen shot has been taken from Protégé[5], a free open source ontology editor and knowledge-base framework. The visualisation itself is part of the Jambal-

[2] Some definitions of ontology depend on an explicitly defined lexicon which provides signs and lexical references. Our definition implicitly assumes that a shared and approved lexicon exists that we can agree on.

[3] http://protege.stanford.edu/ontologies/ontologyOfScience/ontology_of_science.htm, (30.10.2009)

[4] http://ontobroker.semanticweb.org/ontos/ka2.html, (30.10.2009)

[5] http://protege.stanford.edu, (30.10.2009)

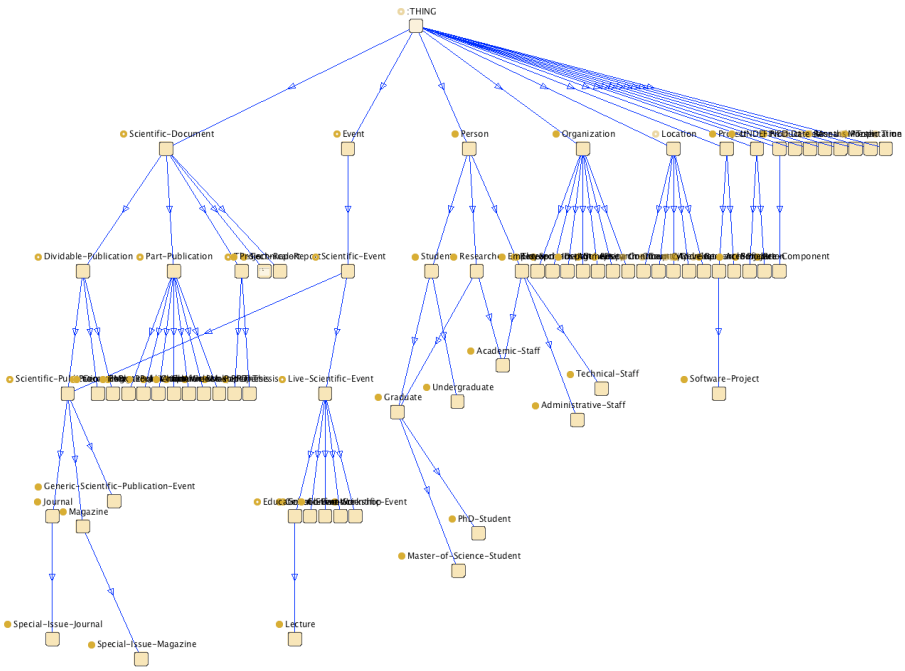

Figure 2.1: Visual representation of the class tree of the science ontology, generated with the Jambalaya plug-in of Protégé.

aya[6] plug-in, which uses Shrimp[7] to visualise regular Protégé and OWL knowledge bases.

Core features of Protégé are class browser and class editor to navigate and modify ontologies. Figure 2.2 shows a partially collapsed class tree on the left side and the property definition of the selected item on the right.

A class entity gets aligned in an *is-a* relation of the class hierarchy, inheriting all properties from its predecessors and extending them by specialisation. Relations to other classes are constraints to be satisfied and must conform with the partial order: Consider a relation between class A and class B, then all subclasses of A inherit a relation to class B or a subclass of B.

The level of detail of an ontology increases with the number of classes. The more classes an ontology has, the more nodes are available for instances to be connected to. However, more specialised classes do not necessarily increase semantic expressiveness. They only increase the level of diversification. Semantic meaning tough

[6] http://webhome.cs.uvic.ca/~chisel/projects/jambalaya/jambalaya.html, (30.10.2009)

[7] http://www.thechiselgroup.org/shrimp, (30.10.2009)

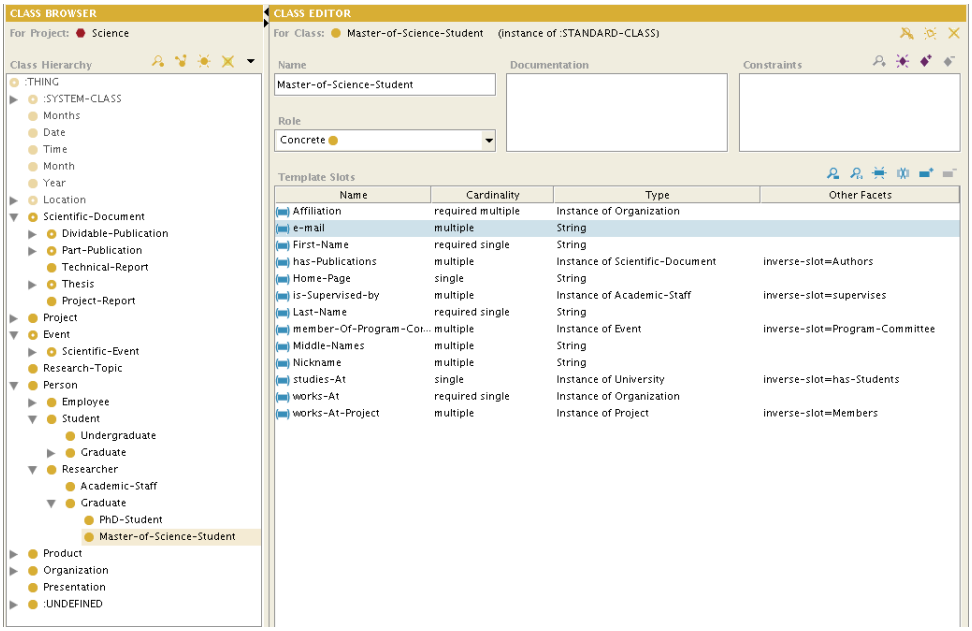

Figure 2.2: Protégé has built-in features to navigate and edit ontology classes

is primarily encoded in the properties and relations of the classes. Thus expressiveness of an ontology increases with the number of properties.

2.2.2 Applications

Ontologies gained a wide acceptance beyond the domain of theoretical sciences and tend to acquire a critical role in e-commerce and business models. They pose as a vital concept for browsing and searching techniques, ensure interoperability in knowledge management, information retrieval and configuration, and deliver the key technique for global controlled vocabularies. These features are extensively used by online catalogues, databases, web publications, and knowledge management applications. McGuinness [2002] provides a comprehensive overview of areas of use of ontologies including controlled vocabulary, site organisation and navigation support, extendible upper level ontology, browsing and search support, disambiguation support, consistency checking, interoperability support, validation and verification testing, configuration, and exploitation of generalisation/specialisation information.

A well-defined ontology, characterising a complete model of a specific domain, is a semantically enriched repository making it very useful for various tasks and applications. Collaborating research groups take advantage of distributed ontologies to share a common, formally defined understanding of parts of the world. Web

resources and applications using harmonised ontologies may depend on a shared vocabulary and a well-defined specification. Software agents can interact and negotiate with each other and traverse the Semantic Web autonomously. A common, formally defined understanding enables them to exchange queries and extract knowledge, which broadens their own knowledge base. Successful navigation and interaction relies on a mutual agreement on the used vocabulary.

The reuse of domain knowledge proves to be a key feature of ontologies. Whereas the distribution of an entire ontology supports interaction between various applications, it is convenient to split up ontologies into separate sub-domains if a simpler task is at hand. Smaller ontologies are easier to implement and to maintain. They can be distributed and managed by collaborating ontology engineers. Merging and integration of various ontologies are well-known methodologies [Gangemi et al., 1998; Pinto and Martins, 2001]. Small ontologies that describe portions of a larger domain can be merged to form a larger ontology. Coarse-grained ontologies are extended to describe a more detailed domain of interest.

The popularity of ontologies increased massively in the last decade. Their simplicity and expressiveness made them a popular subject in scientific research and application development. In recent years, many applications in various research disciplines have been developed to exploit ontologies as an universal knowledge base. Disciplines, such as biology and genetics [Seki and Mostafa, 2005], medicine [Gonçalves et al., 2009], Semantic Web services [Wang et al., 2008], ubiquitous computing [Christopoulou et al., 2005], legal information systems [Mommers, 2003], multi-agent environments [Li et al., 2005], and self-management systems [Zhou et al., 2007], describe the appropriate domain of interest using ontologies.

Building an application based on an ontology supports the separation of domain knowledge (i.e. the underlying ontology) from the operational knowledge (i.e. the algorithms written to perform a task). The operational knowledge can be implemented independently of the domain. If the application logic is strictly separated from the domain at hand, the application can easily be reused to deliver appropriate results on a different domain by simply exchanging the underlying ontology. Another benefit going along with the separation of domain and operational knowledge is the avoidance of domain bias compiled into the application. An application strictly to be build for a specific domain-dependent task induces domain assumptions to be hard-coded. Whereas using an ontology as the solely resource of domain knowledge, all domain assumptions have to be explicitly defined within the ontology. Consistent separation of domain and operational knowledge enables the application to easily adapt domain assumptions within the domain ontology, and avoids domain biased application code.

In Information Engineering relational databases are traditionally used as data storage. Relational databases are characterised by static schema descriptions, allowing users to easily add new data, which conform the predefined structure. Data sets of

unknown structure are difficult to introduce. Usually both database definition and task logic have to be adapted accordingly. The corresponding application is tightly built on top of the specific database layout, implementing domain constraints and assumption, which cannot be encoded within the knowledge base. Newly introduced or adapted constraints are difficult to implement as they have to be applied both to the data storage schema and most likely to the application logic as well.

Ontologies enforce a strict separation between data schema and actual instances. The core ontology schema formally describes the properties and relations that hold for any individual data set within the domain of interest. This schema can then be filled with actual instances of the classes defined by the schema. However, the application built on top of the ontology must not implement any assumptions of the real world data represented by actual instances, but try to reduce domain assumptions and unintended domain and encoding bias to a minimum [Gruber, 1993; Gómez-Pérez and Benjamins, 1999]. In classical Software Engineering there is no explicit separation between operational and domain knowledge. Applications are written both domain- and task-dependent, rendering them useless for deviant assignments and subjects. The use of ontologies encourages the segregation of domain knowledge from operational knowledge. Applications that are following this paradigm tend to avoid domain bias and promote re-usability.

2.2.3 Ontology Engineering

Ontology Engineering is the task of designing, implementing and maintaining ontologies and ontology-based applications — it is an ubiquitous task in the life cycle of an ontology. “Ontologies, as any engineering development, need the definition and standardization, of a life cycle and methodologies and techniques that drive their development." [Gómez-Pérez, 1994]

The task of crafting a new ontology for a specific domain is very time consuming. The development process is necessarily iterative and subject to continuous re-evaluation. Every iteration of revision and refinement compiles a more detailed conceptualisation of the domain. Similar to knowledge engineering the development has to be attended by domain experts [Gómez-Pérez, 1994] sharing a mutual agreement on devised aspects. The resulting ontology is supposed to be an objective description of the concepts in the domain. It is important to understand, that compiling a complete ontology for even a limited aspect of the world is not feasible. The level of detail an ontology provides needs to be limited by a definition of scope and purpose.

Ontology engineering requires thorough knowledge of the appropriate domain and the task the ontology is developed for. Furthermore, all users of the ontology have to agree upon the specifications of the domain described by the ontology. It is obvious

that this assignment needs both sufficient technical and domain knowledge, as well as cooperative skills to satisfy all requirements.

A key motivation for introducing an ontology is the possibility of knowledge sharing. The high costs of knowledge acquisition and formalisation can be distributed among multiple applications and tasks. This reasonable approach competes with the application-driven approach most ontologies have been developed. The resulting ontology usually shows a trade-off between the two approaches. In order to perform reasonable for a particular application the underlying domain knowledge incorporated into the ontology is strongly affected by the nature of the task and purpose. This dependency usually indicates a flaw in the separation of domain and task knowledge. A good overview about the discussion can be found in Guarino [1997].

In order to clarify the differences between domain and task knowledge consider the following example: In computer science the term $P = NP?$ may easily be identified as the still open question whether the class of nonpolynomial problem lies within the class of polynomial problems or not. However, for a computer program to come to the same conclusion, the following steps have to be taken first:

1. Identify a mathematical expression: The program has to have knowledge of the concepts of equations, algebraic notations and the usage of variables.

2. After the successful identification of the presence of an equation it still has to consider the context of the term. A potential (mis)interpretation may be the definition of an algebraic neutral element, where P denotes an instance or variable, and N represents the neutral element. This disambiguation can only be confirmed by specification of the domain of interest and the knowledge of problem classes within this domain.

3. Identify the instantiation of the variables: In this case the equivalence of P being the class of polynomial problems and NP as the class of nonpolynomial problems.

4. Finally, additional knowledge about the history of the problem, the definition of completeness, and the implications of a possible solution should be considered to complete a formal understanding.

The conclusion, that $P = NP?$ represents the question whether the class of nonpolynomial problem lies within the class of polynomial problems, can only be derived by invoking a lot of domain specific background knowledge. Now consider an ontology to explicate the concepts of $P = NP?$. When would you consider it to be complete? Which classes are necessary? There would probably be classes for P, NP, problems, and completeness, but the decision on introducing classes on all algebraic concepts would heavily depend on the purpose and task of the ontology. If the domain of interest proves to be used for formal verification of problem classes,

then probably historical references should be ignored. However, if we conclude that the steps above are crucial for the understanding of the term $P = NP$?, then all concepts have to be explicated in the ontology — including historical references.

This simple example should illustrate that the vital task of an ontology engineer is to cope with the trade-off in the compilation of an appropriate granularity of details. Significant parts of domain expertise are highly implicit and are based on background knowledge. The role of an ontology engineer is to explicate implicit knowledge and compose a task-complete ontology. Whereas a maximum level of granularity of an ontology is to be preferred, unfortunately most ontologies have been constructed with a task-driven approach: Only those parts of the domain that are essential to a given task, are explicitly defined. Therefore the resulting ontologies are difficult to reuse. Domain assumptions not explicated within the ontology have to be explicitly incorporated into the task knowledge.

In order to enhance knowledge sharing and the reuse of ontologies, they have to be crafted to capture background knowledge in a way to make implicit assumptions transparent. Often the level of detail and semantic annotations has to be increased.

If we happen to have ontologies, each of them rendering a complete model of the necessary knowledge about one of the four steps described in the $P = NP$? example, we could compose a merged ontology to conclude the interpretation of $P = NP$? without the need of implicit background knowledge.

Notice that this is a fundamental shift of approaches in information engineering and information retrieval. Traditional information systems usually assume, that the user has sufficient knowledge about the problem domain, leaving the user to cast adequate interpretations. Web search queries, for example, are limited to complex pattern search matches, usually ignoring the context of interest. A search query for "$P = NP$?" probably yields results from many different subjects, hopefully including the intended scope of the issuer. However, the filtering of the results — that is, to find those results that match the intended meaning of the initial search query — is left to the user. In order to be able to identify valid results, the user must have sufficient knowledge of the domain *before* s/he initialises a search. The user needs to have appropriate background and specialist knowledge to be able to work with the information system. Intelligent information support systems are supposed to take responsibility of the knowledge domain, taking the need for detailed domain knowledge away from the user. Ontologies are a promising candidate to help achieving this.

2.2.3.1 Ontology Development

So far, we have introduced ontologies as a key methodology to arouse semantic meaning to knowledge bases, information retrieval, and artificial intelligence. In

this section, we shall discuss the different approaches to create and maintain ontologies.

> "Ontology development can be considered a type of software development and thus should follow the general principles of the software project life cycle." [Obrst et al., 2003]

There are usually two options to create an ontology. Either it is built from scratch, or it is composed from existing ontologies. Ontology building, which refers to the task of creating an ontology from scratch, has been thoroughly discussed in Pinto and Martins [2004]. As a hands-on documentation, Noy and McGuinness [2001] compiled an excellent step-by-step guide to initially design, implement and maintain ontologies.

Different methodologies have been proposed and evaluated for this task. According to Pinto and Martins [2004] all of them can be divided into five subtask: specification, conceptualisation, formalisation, implementation, and maintenance.

Specification Identify the purpose and scope of the ontology. The goal of this task of limitation is to outline the scope and detail of the ontology. The application or task the ontology is built for must be provided with sufficient knowledge to perform reasonably. For the sake of performance, the ontology should not contain gratuitous details the application is unable to use for its task. Notice that this task of limitation imposes penalties for subsequent reuse of the ontology.

Conceptualisation Describe a conceptual model of the ontology to be built, such that it meets the specification. The resulting model consists of concepts and relationships among those concepts.

Formalisation Transform the conceptual description into a formal model. This step yields a more detailed concept than the conceptualisation model. The resulting form can be seen as a preliminary step towards the final formalisation.

Implementation Implement the formalised ontology in a knowledge representation language.

Maintenance Update and correct the implemented ontology. That is, review the classes and relationships among them, and correct possible inconsistencies. Depending on the used representation language, a reasoner can be applied to infer formal errors.

The life cycle of an ontology usually does not end after its creation. Evaluation and maintenance are crucial tasks for an ontology to remain accurate for a given task.

An ontology evolves over time: New knowledge will be incorporated, new classes added or altered to adjust the representation of the domain as the corresponding real world objects change.

All of these modifications have to be evaluated and documented. The task of maintenance interjects the principle of versioning to ontology engineering [Kauppinen and Hyvönen, 2004]. There is no inherent technique to monitor and track the differences between multiple versions of an evolving ontology. Especially in a distributed environment, where collaborative development of a single ontology has been pursued and every party maintains local copies, the principles of versioning have to be obeyed. Keeping track of the various versions of an ontology is an important task of an ontology engineer.

Ontology building is a time-consuming and error-prone task. The resulting ontology is usually limited in expression and detail, hand-crafted to serve a specific application. Reusing ontologies can yield faster results if composed from validated and approved sub-ontologies [Uschold et al., 1998; Simperl, 2009].

There are two concerns that can be faced with ontology reuse:

Ontology Merging Two or more ontologies on the same subject are merged into a larger, comprising ontology by unifying knowledge from the source ontologies.

Ontology Integration Building a new ontology by assembling available ontologies on different subjects. The resulting ontology covers a wider domain than any of the source ontologies.

Both ontology versioning and, more obviously, ontology reuse depend to a great detail on the premise of identifying similarities between ontologies. In the setup of versioning the problem of tracking the evolution of single classes can be faced by identifying the corresponding classes in both versions of the ontology and documenting its development. In ontology reuse the task of finding overlapping classes in different sub-ontologies in order to merge them is based on the successful identification of identical classes. *Ontology mapping* is the methodology to face this problem.

2.2.3.2 Ontology Mapping

Ontologies have been developed for a wide range of domains and for a wide range of applications. Some of the ontologies share a common domain but, as they have been developed independently for different tasks, are not compatible by means of interchangeability. The applications are unable to use different ontologies while remaining accurate on solving the given task. Above that, the applications are unable to interoperate with each other, as they do not share the same terms. Because their

underlying ontologies differ, we can not assume, that they inherit the same meaning for the same classes. Even if they do have the same vocabulary (i.e. the set of class names overlap) the semantic meaning of the classes may differ.

Nevertheless, we can safely assume that, as the ontologies share the same domain, vital parts of the ontologies describe the same part of the world. Thus even if the syntax may differ (e.g. class names) the semantic meaning (e.g. properties of the corresponding classes) are similar. If we could find a mapping of equivalent entities in both ontologies the corresponding applications may interoperate with each other.

The task of identifying relationships between the entities of two different ontologies is called *ontology mapping*. That is, given two ontologies, O_1 and O_2, for every entity in O_1 find a corresponding entity in O_2 with the same or closest semantics. The degree of mapping results can include simple one-to-one correspondences between each entity, or result in a complex declarative mapping. An ontology mapping presents semantic relations between ontologies. Furthermore, not only equality relations can be identified, but more distinguishable mappings, stating for example, that two entities are semantically similar to each other, or that one entity is semantically broader or narrower than the other. In certain configurations, also inequality or opposition are interesting findings, especially when annotated with a corresponding confidence factor. Notice, that mappings usually can only be applied in one direction. We discuss mutual relations as intrinsic part of ontology alignment (Section 2.2.3.3).

The mapping results are usually stored separately from the original ontologies to be provided for further ontology engineering tasks. The focus of ontology mapping lies in the representation to deliver necessary information about the relations between two ontologies. These findings can then be processed by a subsequent task.

In the context of heterogeneous ontologies and the need of knowledge sharing ontology mapping is a crucial activity to facilitate interoperability. To stress the importance of ontology mapping, we shall discuss multiple applications of ontology mapping [Euzenat and Shvaiko, 2007].

Ontology engineering. We already discussed in detail the necessity of ontology mapping in ontology engineering. Ontology merging and ontology integration depend greatly on the ability to match overlapping entities from different ontologies. The identified entities can then be used to incorporate the input ontologies, building a larger ontology on top. Ontology reuse can hardly be obtained without the utilisation of ontology mapping results.

Ontology evolution and versioning are mastered by computing a mapping of multiple versions of the same ontology. The mapping result can be appreciated as semantic difference between evolving ontologies.

Information integration is the task of integrating diverse information from multiple resources and of different representations. The user is then presented with

a harmonised interface for query input, rendering the heterogeneous layout of the underlying resources transparent. In the background, however, the query engine has to interpret and rewrite the initial enquiry to match the query language of the appropriate information source. The translation is accompanied by a mapping step to identify correspondences between semantically related entities among different data sources. Before presenting the user with the final answer, the results obtained from multiple information sources are reconciled and filtered for redundancies and duplicates.

Peer-to-Peer Information Sharing. "Peer-to-peer (P2P) is a distributed communication model in which parties (also called peers) have equivalent functional capabilities in providing each other with data and services [Euzenat and Shvaiko, 2007]." Traditional P2P systems provide rather strict schema restrictions, allowing the user to tag their data only with elemental meta information (e.g. author, title, file size, ...). Every peer has to adapt to the usage of the global schema, being unable to modify or extent it. Although peers are granted autonomy in respect to participation, they are denied to act autonomously in respect to schema design. In more advanced semantic P2P systems this flaw has been tried to overcome by relaxation of homogeneity requirements. Peers are allowed to introduce more complex specifications of their concepts by providing ontologies to describe content. Along with the introduction of different ontologies emerges the necessity of mapping equivalent concepts between independent schemas.

Web Service Composition. Web services are platform-independent software components that are available in the distributed environment of the Internet. Based on this paradigm applications are assembled by composition of appropriate web services rather than programmed manually. Web service processors are supposed to deal with independent and replaceable web services to achieve a particular goal. The problem of web service composition [Srivastava and Koehler, 2003] is to parse and compare the descriptions of promising services, incorporate and dynamically choose new services, and to appropriately route the data they process. To compose different services, data has to be routed from the output of some services to the input of another service. If the underlying ontologies of the connected services differ, they have to be mapped so as to enable knowledge sharing.

Autonomous communication systems. When two autonomous and independently designed agents meet in a multi-agent environment, they have little chance to negotiate with each other if they do not share the same ontology. "Agents confronted with heterogeneous ontologies have to find the correspondences between these ontologies in order to start understanding each other's messages [Euzenat and Shvaiko, 2007]."

In ambient computing a similar problem arises if mobile devices try to take advantage of the surrounding environment for providing services to users. The need of alignment of ontologies between personal agents and environmental devices is a preliminary requirement for a mutual understanding and initial negotiation. Above that, agents have to cope with environmental changes and keep track of newly appearing or changing devices and sensors.

Navigation and query answering. In an open distributed environment all communication relies on a mutual understanding among all participants. Navigation within such an environment — as in the Semantic Web — strongly depends on the agreement of common knowledge. Unlike in information integration, where the use of a global knowledge schema is obliged, the Semantic Web does not provide such a convenience. Every interaction between two participants is preceded by a negotiation of the communication schema. In the traditional World Wide Web users are accustomed to use their own terminology for queries. Semantic query answering systems transparently rewrite the query according to available ontologies and translate it into the language of the appropriate query system.

Considering the topology of the Web today, the majority of its content is neither directly linked nor automatically indexable. This part of the Internet is called "Deep Web" or "Invisible Web" and refers to data sources that provide considerable amount of information with back-end databases. Manual query interfaces are the only accessible way to gather information from the Deep Web, thus preventing web crawling engines from automatically extracting and indexing their content. An et al. [2007] outlines an approach to facilitate automated information extraction from the Deep Web by automatic attribute extraction and ontology-based similarity comparison.

Ontologies are a promising concept to model a semantically enriched knowledge base that provides information about a specific domain of interest, to be used by a specific application. The lack of a single, globally agreed on and omnipotent ontology, rendering every aspect of the world, makes ontology mapping an indispensable technique to introduce knowledge sharing and reuse. Thus, the power of ontologies can only be unleashed by advancement in ontology mapping tools.

2.2.3.3 Ontology Alignment

So far, we did not pay much attention to the distinction between ontology classes and specific instances, as they do not differ in syntax. It is more a matter of semantics whether the leaf nodes of an ontology are interpreted as instances or concepts. Therefore the ontology engineering methods introduced so far can be applied to ontologies both with and without instances. In this regard ontology alignment proves to be an exception.

Given two ontologies, ontology alignment describes the task where for every entity in one ontology try to find a corresponding entity in the second ontology with the same or closest meaning. The intended purpose of ontology alignment is to bring two ontologies into mutual agreement. The respective ontologies are tailored to be interchangeable and semantically equivalent. For this cause, it may be mandatory to map entities that are specified as classes in one ontology, but implemented as instances in the other ontology. Whereas ontology mapping concentrates on finding mappings between two individual entities, ontology alignment represents a more holistic approach.

A preceding mapping process yields a list of one-to-one correspondences, representing the equality relation that holds between the mapped entities. Obviously, there may be entities that have no corresponding term in the other ontology. In case of such a partial alignment, new concepts and relations have to be introduced to provide a sufficient target in either ontology. In contrast to ontology mapping, in order to complete an ontology alignment, one or more ontologies are adapted and have to be extended by the missing parts of the opposed ontology.

In the alignment of heterogeneous ontologies it proves to be insufficient to consider only equality associations. Real-world ontologies tend to be implemented with different granularities. Multiple entities of an ontology can bear characteristics that are encoded in a single entity of the opposed ontology. Whereas the semantics are described in a single class, the same meaning can be described by union of multiple classes. It is very likely that compared entities are not completely the same, but — by missing vital properties — are related by subsumption or extension.

As an extension of one-to-one mappings, if the alignment relation between two entities fails to be described as equal, try to apply any one of the following associations: syntactically equal, similar, broader-than or narrower-than. The identification of these *complex alignments* adds significant complexity to simple one-to-one identity mapping algorithms.

Notice that a successful match according to the alternative associations does not conclude an alignment. It does, however, alleviate a subsequent manual alignment process. The main focus of this thesis is the implementation of a semi-automated approach to ontology alignment that has been outlined by Lanzenberger and Sampson [2006]. Starting point is a preliminary, computer-generated mapping, which includes a wide range of semantically annotated relations that hold between the entities of two ontologies. Based on these mapping results, an ontology engineer can easily identify missing concepts in either ontology and finalise the alignment process.

2.2.4 OWL Web Ontology Language

The World Wide Web Consortium (W3C) arranged a working group to develop a standard ontology language to be used to share and exchange *meaning*. The result was the OWL ontology language standard[8]. OWL is based on a Description Logic (DL), being a logic-based knowledge representation formalism. DLs implement an object-oriented model, describing a domain in terms of concepts (classes), roles (properties and relationships) and individuals (instances). The key feature of DLs (and other logics) is that they have a formal semantic.

> DLs can, in fact, be seen as decidable subsets of first-order predicate logic, with individuals being equivalent to constants, concepts to unary predicates and roles to binary predicates. As well as giving a precise and unambiguous meaning to descriptions of the domain, this also allows for the development of reasoning algorithms that can be used to answer complex questions about the domain [Horrocks, 2007].

The implementation of OWL uses an RDF[9] based syntax to incorporate seamless into the Semantic Web context.

Ontologies, being designed to be modular and easily extendible, tend to become very large and complex (e.g. the Cyc ontology[10], the National Cancer Institute thesaurus[11], The United Nations Standard Products and Services Code®[12], the Systematized Nomenclature of Medicine–Clinical Terms (SNOMED CT®)[13]). The task of designing and developing an ontology for a domain of interest is very time consuming and error-prone. The complexity increases with the number of classes and corresponding properties. Modern ontology engineering tools assist the engineering process not only with an adequate design and layout, but with built-in reasoning support. Ontologies modelled in OWL represent a logic knowledge base, offering inferring methods to be applied. Reasoning at design time offers the benefit of identifying design errors at an early stage and can be used to verify the correctness of the developed ontology. For example, inconsistencies and synonymous classes can easily be identified. In ontology engineering, an inconsistent class describes an *over-constrained* class where the description of its properties denies the existence of instances. It is usually an unintended feature of the design if a class is designed not to have any instances, therefore a reasoner will indicate such a class and yield a warning. *Synonymous* classes have exactly the same set of instances. Whereas it is sometimes desirable to have alternative class names for the same set of instances,

[8] http://www.w3.org/TR/owl-semantics/, (30.10.2009)
[9] http://www.w3.org/RDF/, (30.10.2009)
[10] http://www.opencyc.org/, (30.10.2009)
[11] http://ncit.nci.nih.gov/, (30.10.2009)
[12] http://www.unspsc.org/, (30.10.2009)
[13] http://www.nlm.nih.gov/research/umls/Snomed/snomed_main.html, (30.10.2009)

it may be a result of unwary property definitions causing different classes to be semantically equal.

As we have seen in Section 2.2, the hierarchical structure of the classes of an ontology is constructed merely by the relations defined by the class properties. Ontology tools implicitly assume that every class is part of the partial order $\leq_C$ we defined in Section 2.2.1. A newly introduced class is defined as a subclass of an existing one, whereas the application handles the proper definition of relationship that holds between the parent and child class. This approach holds the option to integrate implicit assumptions that are not formally represented in the knowledge base. The use of a reasoner may help to avoid such a mistake. The descriptions of the classes are explicitly defined and should be complete and precise to render the hierarchical structure. Thus the computation of the class hierarchy can be left to the reasoner. The ontology engineer afterwards verifies whether the result is consistent with the intended ontology. If the resulting hierarchy does not comply with the intended meaning, the class descriptions have to be adjusted.

In order to work with or develop large ontologies it may be useful to modularise the knowledge base. A reasoner can help identifying valid subsets and dividing an ontology into smaller modules. This technique can be used to facilitate parallel work or create subsets of the domain to be used by other applications. Additionally, smaller ontologies are easier to comprehend and verify. With the help of ontology integration the final ontology can be derived.

2.2.5 Ontology Alignment Representation

Heterogeneity proves to be a major problem of the Semantic Web. Ontologies have been deployed to cope with this issue. Unfortunately, due to their independent integration, ontologies themselves tend to introduce inhomogeneity when various ontologies cover similar subjects. This problem has been approached with ontology alignment, trying to align ontologies with each other and enable interoperability of applications using different ontologies. Again, the individualistic development of various ontology alignment methodologies calls for reconciliation on alignment representations to increase interoperability between ontology mapping and ontology alignment tools, and to bundle collaborative effort in the development of ontology alignment tools.

The standardisation of ontology alignment representation initiates central features for the Semantic Web community:

Archiving. The task of ontology mapping is usually a preliminary step in a more complex ontology engineering work flow, making the mapping results volatile. However, the mapping representation may be preserved for later reference in order to reconstruct the mapping process. Especially in ontology

evolution and versioning, an archive of ontology mappings an ontology has gone through can be analysed to gain information about ontology evolution.

Distribution. In collaborative environments the impact of a specific ontology mapping must be discussed with participating ontology engineers. For this matter a concordance on the exchange format has to be established. An agreement on the representation of such ontology mappings will improve collaboration quality among distributed parties.

Comparison. In the context of a specific ontology alignment task, the choice of a specific ontology mapping algorithm among various competing contributions, proves to be crucial to the overall performance. A standardised output representation of mapping results facilitates direct comparison between different ontology mapping algorithms.

Modularisation. With a standardised interface for ontology mapping results, ontology authoring tools can be modularised to implement different adaptors for exchanging results and provide input to various consecutive tasks.

Driven by the crucial need for ontology mapping tools, several different independently developed methodologies have been introduced. Even though the problem is very well specified, the actual implementations of the alignment results differ significantly. During the work on AlViz, we discovered that no appropriate ontology alignment representation standard exists, that covers the specific requirements of the task sufficiently.

An appropriate ontology alignment representation format has to meet the following qualifications:

Representation of detailed alignment features. AlViz initially used the FOAM mapping algorithm as input for the visualisation, but during the implementation, the authors had future development in mind, assuming that various mapping tools can be incorporated. However, compared to other ontology alignment techniques FOAM delivers a very detailed alignment feature set, including 24 similarity functions and confidence factors for every mapping. This rich feature set has to be properly encoded in the specified format.

Human and Machine Readable. The output format is supposed to be machine-readable, for obvious reasons, as the ontology mapping results are usually an intermediate step in a complex ontology engineering process and are supposed to be machine processable. Additionally, the demand for a human-readable format comes from various considerations. First, as an development and integration aid: Developers can easily verify whether their algorithms correctly generate, parse, and transform mappings. Second, for a quick glance, users can look at the raw file to search and identify nodes and

their corresponding mappings. Finally, in accordance to the semantic web paradigm to provide semantic content and enable sensible navigation for both human and machine agents.

Verifiable Schema specification. Another requirement is the need for formal specification of the file schema. The alignment format has to be well defined and verifiable. The requirement is easily satisfied by the use of XML:Schema[14](`XSD`) or Document Type Definition[15] (`DTD`). Both description languages support checking and validation, and can be used to identify valid input files before the actual parsing takes place.

Transformation support. Furthermore, the chosen format has to alleviate transformation of alignment results into various representations and enable subsequent processing. In the case of AlViz, the mappings are filtered and transformed to visualise alignment links between two ontologies.

To our best knowledge, the proposal of an ontology alignment format that resembles the requirements of AlViz closest, has been provided by Euzenat [2004]. Unfortunately, while matching the given requirements in almost all points, the actual schema definition does not allow a degree of detail, we need to express all alignment features. Due to restricted time constraints we finally decided to implement a custom-built specification of ontology alignment format. However, we are pleased to notice that our definition resembles a substantial similarity to the proposal of Euzenat [2004]. We are confident that our approach can easily be implemented as an extension of Euzenat's alignment format or even be merged. As a first step for a consolidated alignment format Gradwohl [2009] extended the Euzenat API in order to suit the requirements of AlViz. The proposed XML format has been successfully implemented and will be used as the primary alignment format in future AlViz versions.

So far, we have discussed the technical requirements and implications of ontology mapping and ontology alignment in detail. But when it comes to formally define and process *meaning*, we have to face limitations that fail to be resolved by technical means. In Artificial Intelligence literature this border is called the *Semantic Gap*.

2.2.6 Ontologies and the Semantic Gap

Every attempt of transforming a real world object into a formal computational representation induces a semantic disparity between the real world object and its description. For the human mind an object is only perceived as an interpretation regarded on the contexts of both, the object and the human being itself. The translation into

[14] `http://www.w3.org/XML/Schema`, (30.10.2009)
[15] `http://www.w3.org/TR/REC-html40/sgml/dtd.html`, (30.10.2009)

natural language, and subsequently into a formal language to be represented by a computer, is thus accompanied by a loss of expressiveness. This characterisation is dubbed *semantic gap* or *semantic mapping problem*.

> The semantic gap is the lack of coincidence between the information that one can extract from the visual data and the interpretation that the same data have for a user in a given situation [Smeulders et al., 2000].

Whereby the current centre of interest of ontology mapping research concentrates on the development of automated mapping tools to devise reliable mapping algorithms, it should be noted, that a fully automated mapping process can not be achieved. Considering the intended purpose of ontologies, i.e. explicate implicit knowledge of parts of the real world and providing a complete formal model of the domain of interest, this implies a noteworthy deficiency.

The evaluation of ontologies is a major concern in the AI community, however, it concentrates on the technical aspects and implications of the implementation. Even though the purpose of the evaluation has been proposed generously — "The goal is to prove compliance of the world model (if it exists and is known) with the world modelled formally [Gómez-Pérez, 2001]" — the actual validation of requirements and competency questions are performed qualitatively. That is, a group of human experts assert the quality of compliance.

Ontology mapping is a hard problem in Artificial Intelligence, dealing with human notions of knowledge and the comparison of semantically annotated entities. The categories of possible mismatches of ontologies have been identified by Klein [2001] and extended by Sampson [2007]. The mismatches can arouse at different levels:

Language Level. Mismatches at the language level occur when ontologies that are described in different languages are compared with each other. These conceptual languages may differ in the specifications of their underlying paradigms, representation of logical notions, or their level of expressiveness. Given thoughtful consideration, this kind of mismatches can usually be solved automatically.

Ontology Level. Ontology level mismatched can occur when two ontologies are expressed in the same language, but tend to have different conceptualisations of the same domain. The characteristics of mismatches on an ontology level include the scope of concepts, model coverage and granularity, paradigm differences, competing modelling conventions of concept descriptions, synonym and homonym terms, and encoding disparities. In a nutshell, mismatches on the ontology level occur, when two different ontology engineers build two different ontologies in the same domain, using different terms

(or granularities) to describe the same concepts, or use the same notation to describe different concepts.

Human Level. Human level mismatches are defined by Sampson [2007] as the differences of (biased) views and interpretations of the real world among communities, groups, or single ontology engineers. "[T]hey represent a 'model' based on that subjective view of the developers and not the model of that part of the real world being investigated." She continues, "We claim that differences among the way we perceive, understand and articulate constructs affect the way we create ontologies. These differences often bring about the need for alignment of ontologies in the first place."

Even though Klein [2001] identified the style of modelling in his classification of mismatches as a source of disparity, he failed to give appropriate tribute to the fact that many mismatches are based on different human interpretations. Even though he states, that "the alignment of concepts is a task that requires understanding of the meaning of concepts, and cannot be fully automated", we believe the level of human mismatches to be the main cause for the lack of fully automated ontology alignment techniques.

Although significant progress has been made in the development of automated ontology mapping tools to provide promising results, a fully automated algorithm is not feasible. In the long run, the problem we are facing in ontology mapping, narrows down to the question, whether computers will — or even should — ever be able simulate human perception and understanding.

2.3 Visualisation

Visual images have an ancient history of helping people explaining complex phenomena. When the first caveman chiselled the location and direction of an approaching antelope herd onto a cave wall he probably used a vivid visualisation to distribute his knowledge. Legend has it that Archimedes died for his circles he was drawing on the sand (*Noli turbare circulos meos!*). Leonardo da Vinci used graphic visualisations to describe his inventions, making people understand and recognise his genius mind.

The human mind undoubtedly processes visual data faster and more reliable than any other sensual input. Perception and cognition are closely interrelated, rendering the human visual system into a pattern seeker of enormous power.

The old proverb "a picture is worth a thousand words" analogously illustrates what only recently has emerged into a young scientific research area: Information Visualisation (InfoVis) is "the use of computer-supported, interactive visual representations of data to amplify cognition." [Card et al., 1999] The crucial property of

this definition are the last three words “to amplify cognition”. Thinking and understand does not entirely occur inside our brain. “Most cognition is done as a kind of interaction with cognitive tools, pencils and papers, calculators, and increasingly, computer-based intellectual supports and information systems.” [Ware, 2004] The human mind is able to increase its cognitive potential if it is supported by external artefacts of stored intermediate data. Such external objects provide an additional form of information representation that can be directly perceived and used without being interpreted and formulated explicitly [Zhang and Norman, 1994].

> “The core of the benefits provided by visuals seems to hinge upon their acting as a frame of reference or as a temporary storage area for human cognitive processes. Visuals augment human memory to provide a larger working set for thinking and analysis and thus become external cognition aids.” [Fekete et al., 2008]

Information Visualisation aims at providing such visual artefacts and developing insights from collected data for decision making support. “One of the greatest benefits of data visualisation is the sheer quantity of information that can be rapidly interpreted if it is presented well.” [Card et al., 1999] However, it is important to note, that graphical representations may promote — but as well — hinder information processing tasks [Ng, 2000].

Ware [2004] highlights a number of additional advantages of visualisation.

- Visualisation provides an ability to comprehend huge amounts of data.
- Visualisation allows the perception of emergent properties that were not anticipated.
- It is common for a visualisation to reveal things not only about data itself, but about the way it is collected thus providing invaluable quality control.
- Visualisation facilitates understanding of both large-scale and small-scale features of the data. It can be especially valuable in allowing the perception of patterns linking local features.
- Visualisation facilitates hypothesis formation.

In ontology alignment visualisation we will strongly depend on this properties and develop and apply visualisation techniques that suit the needs of the task.

2.3.1 Ontology Visualisation

The availability and propagation of ontology description languages introduced reuse and portability to ontologies. Various tools and applications have been developed to facilitate ontology authoring, manipulation and distribution. Many automated tasks and applications based on OWL have been developed, proving its reliability as computer-readable language. Unfortunately, the same ontology representation lacks human-readable processing. The larger an ontology becomes the more likely a human reader fails to conceive its semantic meaning by just analysing its OWL representation.

This lack of human perception weights heavily on the subject of ontology research. Still, the need of a human analyst for given or developing ontologies remains to be essential for a successful implementation. The concept of ontologies is based on semantic perception and agreement of parts of the (human) world. Thus the representation and implementation of such a model (i.e. the ontology) can only be evaluated by a human expert, trying to bridge the semantic gap. It is therefore inevitable to create a form of representation that is easily perceived and adapted by a human but which simultaneously maintains the feature of effective parsing and evaluation by a computer.

The size and complexity of ontologies render mark-up languages useless for this task. It seems that there is a fundamental trade-off between computer and human-readable concepts of representation. Lines and lines of repeating tags and attributes are easily processed by an automated parser, leaving a human reader on the other side to get lost in the repetitive source code.

It is common practice in database engineering and information systems to use diagrams and visualisations to present the content and structure of the analysed system (e.g. Entity-Relationship (ER) diagrams [Chen, 1976], Unified Modeling Language (UML)[16]). The key features of such visual representations are analogous visualisation of vital properties in graph theory, encoded by attributes like vicinity or size. The entities placed on the visualisation plane are tagged according their mark-up properties, whereas dependencies between symbols identify relationships. Similar visualisation techniques have been developed and applied to ontologies, leaving the ontology community with a variety of different visualisation tools.

Along with the increasing usage of ontologies emerged the need for visualisation methods to alleviate their design, management and analysis. More specifically, a survey initiated by Ernst and Storey [2003] identifies the following use cases for an ontology visualisation tool:

- Assist with navigating information space
- Show hidden relationships

[16] http://www.uml.org/,(30.10.2009)

- Show areas of interest
- Check for inconsistencies or errors
- Present reports to others
- Help with understanding new ontologies
- Identify and describe relationships

There are several ontology visualisation methods discussed in ontology literature. Katifori et al. [2007] assembled an excessive survey on various ontology visualisation techniques and categorises their characteristics and features. Their conclusion, however, leaves us unsatisfied: "there is not one specific method that seems to be the most appropriate for all applications and, consequently, a viable solution would be to provide the user with several visualizations, so as to be able to choose the one that is the most appropriate for his/her current needs." That is, even though there exists a variety of visualisation techniques for ontologies, no golden standard can be named: There is no specific visualisation method to be the most accurate for any application. Every single visualisation technique yields specific assets and drawbacks. A viable solution to this dilemma appears in providing the user with several different visualisations. Furthermore, users tend to appreciate visualisations that offer orderly and clear browsing, even if it requires zooming into the ontology and focusing on a specific part. However, browsing does not suffice for tasks involving the location of specific nodes. Users prefer effective search tools or query mechanisms to tedious browsing, especially if large ontologies have to be examined. After all, one of the hardest problems of ontology visualisation remains to be scalability.

> The visualization of ontologies and metadata is a challenging issue with several applications not only in the Semantic Web but also in Software Engineering, Database Design and Artificial Intelligence [Tzitzikas and Hainaut, 2006].

As ontologies show the tendency to grow very large in respect to both, number of entities and amount of detail, the resulting visualisation tends to become cluttered and disorganised.

2.3.2 Ontology Alignment Visualisation

Notice, that ontology visualisation significantly differs from ontology alignment visualisation. Whereas the problem of ontology visualisation has been thoroughly discussed in ontology and information visualisation literature, the field of visual ontology alignment is still a very young discipline. Accordingly, there are very few visualisation tools for assisting with ontology alignment. However, the results of

user experience concerning ontology visualisation can be used as a starting point and contribute valid insight for requirements of ontology alignment visualisation.

In the first survey that tackles the problem of ontology alignment visualisation, Sampson and Lanzenberger [2006] state some interesting findings that distinguishes the requirements of ontology and ontology alignment visualisation. Starting with well-founded issues based on crucial requirements for ontology visualisation, they identified two issues that get intensified when applied to ontology alignment: First, to maintain smooth interaction for the users, time complexity needs to be held low. Interactivity and responsive controls are a vital part of the visualisation, enabling the user to swiftly verify a specific alignment by switching from the alignment view into the detailed view of a single ontology.

Second, the given visualisation space needs to be used more efficiently to present the user with a large number of concepts simultaneously. A global view of the topology of an ontology may indicate specific properties, which can be used to locate similar parts between the aligned ontologies.

Both requirement amendments can be derived from the fact, that in ontology alignment two ontologies have to be observed and analysed, thus doubling the amount of raw input data necessary for the visualisation.

Research in the field of information visualisation [van Ham and van Wijk, 2002] suggests three proposals to challenge the scalability problem of visualising a large number of nodes:

1. Increase available display space, by either using three dimensional and/or hyperbolic spaces.
2. Reduce the number of information elements by clustering or hiding nodes.
3. Use the given visualisation space more efficiently by using every available pixel.

In addition to generic visualisation requirements, Sampson and Lanzenberger [2006] identify supplemental issues that are genuine to ontology alignment. The analysis implies that properties needed for a single ontology raise a conflict when applied to the visualisation of alignment results based on two ontologies. Existing tools for visualising ontologies are designed for a specific range of queries, thus lacking support for visualising certain ontology characteristics needed for alignment. Also, “in order to support an overview and detail approach appropriately, multiple views are needed for ontology alignment.” These findings support the assumption that the requirements of ontology visualisation differ significantly from requirements of visualising ontology alignments.

Visualisation of a single ontology usually has its focus on helping the user to understand the internal structure, the relationships within and the descriptions of the

classes of the ontology considered. The goal of visualising ontology alignments, on the other hand, is to facilitate the identification of similarities in the structures of both ontologies, emphasising the relationships between them, and pointing out the differences of similar classes. The different aims already indicate that a good ontology visualisation may not be a good candidate for visualising ontology alignments, since it would give information about the two individual ontologies in detail, but not about the relationship between them, which was obtained through alignment. The main difference between ontology visualisation and ontology alignment visualisation is that, in the latter, we do not aim for a *correct* visualisation of all details of an ontology, but rather for decent support of a vivid characterisation of the differences and alignments. A good ontology alignment visualisation uses the available screen space to show the relation between the two ontologies, while necessarily hiding details of the two aligned ontologies.

Nevertheless, in order to decide whether two classes of aligned ontologies are similar, detailed information about the class definitions need to remain available. Ontology alignment visualisation has to be implemented as a two-tier approach, including an *alignment view* and a *detailed view* showing the source ontologies. First, the alignments between two ontologies have to be visualised, emphasising the differences and pointing out the similarities among appropriate entities. A global view of both ontologies puts an overview of alignments into display. Starting from this initial screen, the presented ontologies can be explored and compared. The verification of a specific mapping is carried out by examination of the detailed view. This screen allows the analysis of ontology structures, neighbourhood, and class properties. Based on the information propagated by the ontology definition, the mapping can be verified and the alignment finalised.

We can summarise the findings and compile a list of considerations for a successful implementation of ontology alignment visualisation.

- The underlying technical task at hand is a visualisation of multidimensional data covering ontology definition, alignment mappings, and the aggregation of both. The separation of the individual ontologies and their appropriate mappings is essential for a rapid understanding.
- Data complexity increases significantly as compared to ontology visualisation. Scalability and overall performance has to be kept in mind from the very beginning to ensure a decent user experience.
- Visualisation of information is usually obtained following two steps: First, a transformation of the raw data into a preliminary, semantic structure has to be performed. During this step, in order to specify *what* to visualise, the raw information gets selected, filtered, transformed, classified, or merged. Second, the actual visualisation is drawn using the first transformation as input data. The second step concludes the question, on *how* to visualise. Obviously, the

choices selected in the first step limit the options for the second step. In the case of ontology alignment visualisation this means, to thoroughly discuss the necessary raw data and the impact on the rendering performance.

- For a successful ontology alignment visualisation information about the ontology mappings has to be obtained and augmented with complete information of both ontology definitions. The information sources (i.e. the raw ontology data, and the alignment information) should be stored separately, or at least be easily extracted and distributed.
- The introduction and utilisation of an open and well-defined alignment format alleviates cooperation and exchange.
- The user should be presented with separate views for each ontology and the ontology alignment visualisation. A seamless transition between both views facilitates navigation within, and observation of details of specific areas.
- Interactivity is a core feature of the visualisation. A seamless user-induced transformation of the graph is vital to the understanding and utilisation.
- Performance is task critical and has to be considered from the beginning.

The basic setting of an ontology alignment visualisation constitutes of two ontologies and a list of mapping results. The ontologies are represented in a static OWL structure, which has to be preserved for subsequential automated processing. The mapping results can be derived from an automated alignment process and are exploited for a preliminary visualisation. One approach to cope with this task has been presented by Lanzenberger and Sampson [2006], introducing a methodology to maintain the OWL structure in order to retain automated parsing of the ontology. As an extension, the very same OWL document is used to introduce a visualisation of the ontology, which can easily be perceived by a human user. The ontology visualisation is primarily tailored to facilitate alignment of two given ontologies. Thus, the key motivation is to enhance similarities and differences between the compared data structures. The necessity to render the ontologies formally correct can be reduced as long as it assists the comparison process. Hence, the formal representation of an OWL document can be exploited for automated processing, whereas the graph visualisation facilitates a human ontology engineer to comprehend the differences and similarities of the aligned ontology structures.

2.3.3 Graph Visualisation

Based on the inherent structure of an ontology, i.e. emerging and cascading from a single root node, the intuitive visualisation approach would imply some sort of tree graph. It is therefore not surprising that many ontology visualisation tools include

a tree visualisation. Luckily, the research in ontology visualisation methods has been intensified in the last decade, resulting in various different approaches and techniques to chose from [Katifori et al., 2007].

The classic explorer-like tree visualisation can still be found in many ontology authoring tools. Whereas this visualisation provides an indispensable aid for navigation and search within a given ontology, it yields some problems concerning its accuracy. The tree view primarily displays inheritance (*is-a*) relations. Multiple inheritances are usually resolved by placing these nodes under all its parents. This approach implies difficulties for inexperienced users, who may be confused as to why a class appears multiple times in different sub-trees of a single ontology. However, the main concern about the tree view visualisation arises with more complex ontologies where role relations are specified. The inheritance relations are indeed perfectly covered by the tree view, but the visualisation method does not provide means to represent role relations. To be more specific: The tree view can only represent one kind of relation, whereas general ontologies can comprise an infinite number of interrelations.

This unbound property of ontologies contributes to the intensified search for proper visualisation techniques. It is primarily responsible for the conclusion that no single visualisation method can be applied to cover all aspects of an ontology. It is therefore not possible to map the vast amount of different ontology structures to a specific graph topology. In a nutshell: If we intend to render a graphical model of a random ontology, the properties of the resulting graph will not be known to us a priori. The schematic restrictions of the ontology definition do not confine the graph topology.

In ontology visualisation we will be confronted with graph topologies varying from simple trees to random graphs. However, this is true only if we restrict the link properties of the visualisation to be based on the inheritance relation solely. By expanding the possible link coverage to include all possible role relations of an ontology, the resulting graph topology will probably resemble a small world graph [Gil and García, 2004].

The search for a visualisation method that works best for any given ontology must therefore be approached from a graph theoretical angle. A node and link graph constitutes a connection topology that usually can be defined by two vital properties: The clustering coefficient and the characteristic path length. Many empirical graphs devised from natural phenomena tend to have a large clustering coefficient and a small characteristic path length (also known as average shortest path length). Those graphs are called small world network, by analogy with the small world phenomenon postulated by Milgram [1967]. Many biological, technological an social networks — including the World Wide Web [Adamic, 1999] and the Semantic Web [Gil and García, 2004] — have a small world topology. They are difficult to analyse using conventional means.

Visualisation techniques specialised on small world graphs can cover a wide range of different graph layouts, basically because such graphs are neither regular nor random graphs.

Stephen Ingram presented such a visualisation algorithm named Small World Graph Visualization [Ingram, 2005]. An actual rendering of his graph visualisation can be seen in Figure 2.3. His algorithm is an implementation of a technique devised by Noack [2005] to calculate the lengths of edges in a graph based on an energy based clustering model. The problem with a generic physical layout is, that it "seeks to minimize the variance in edge lengths, leading to a uniform graph layout [Ingram, 2005]." In order to render a distinguished and appropriate cluster visualisation we depend on a variety of edge lengths, with shorter lengths between near-by nodes and longer edges between distinct clusters. Noack's algorithm — dubbed *LinLog* — describes such a model in which the edge lengths are proportional to their coupling.

van Ham and van Wijk [2004] introduced a series of interactive techniques for cluster graph visualisation to be applied on top of the node geometry computed by the Noack layout to define a dendrogram[17] of nodes. These technical specifications allow a smooth interpolation between nodes in a given sub-tree. This semantic/geometric interpolation, together with a fisheye focus-plus-content technique, allows the user to explore the cluster hierarchy and reveal the semantics behind the node community.

The use of a clustering small world algorithm for ontology visualisation introduces some promising advantages:

- Cluster visualisations support hierarchical structures by collapsing all nodes of a sub-tree into a single cluster node. The user is presented with a clear view of high-level objects that comprise all characteristics of their specialised child nodes. An overview of these cluster nodes helps to gain a basic understanding of the graph structure and offers a global starting point for both, detailed examination, and opportunistic browsing.

- Energy-based methods for creating line drawings of vertices in undirected networks are popular for visualising node and link graphs. These methods generally consist of two parts: an energy model, and an algorithm to calculate a state with minimum total energy. Earlier algorithms failed to provide shorter vertices between near-by nodes and longer edges between distinct clusters. The Noack algorithm [Noack, 2003, 2005] succeeds in clearly isolating clusters by appropriately calculate the length of vertices between graph nodes, rendering the layout's edge lengths proportional to their coupling. The visualisation instantly reveals the relations between clusters, varying the distance between them based on their semantic similarity.

[17] A dendrogram is a tree diagram that indicates the sequence in which a hierarchical clustering of nodes is performed.

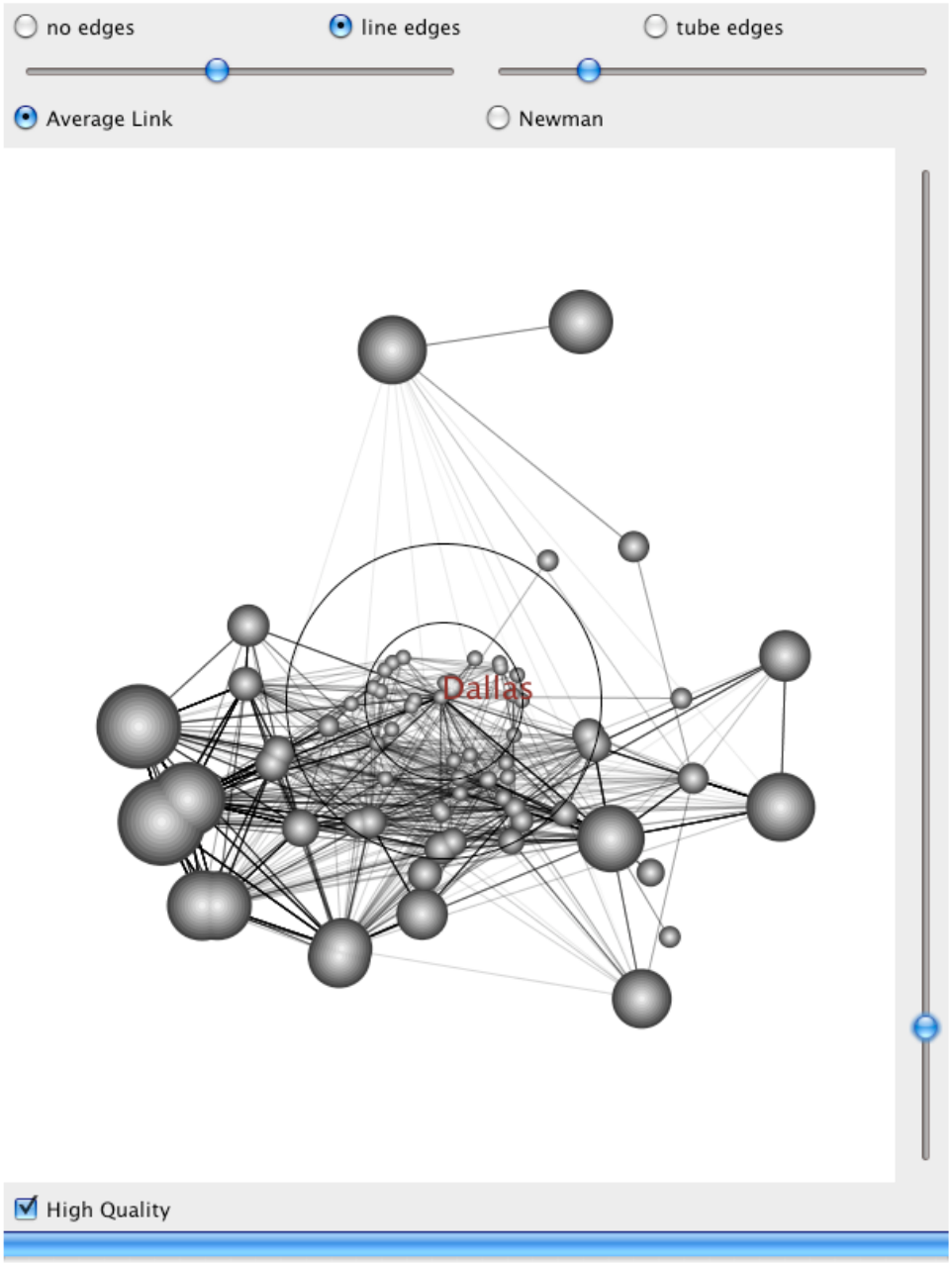

Figure 2.3: Stephen Ingram's Small World Graph Visualization. [Ingram, 2005]

- Many ontologies (particularly medium and large sized) tend to have a high degree of clustering, which makes them ideal for cluster visualisation. An adequate visualisation can efficiently advertise the underlying structure [Lanzenberger and Sampson, 2006]. Furthermore, the specific implementation of clustering visualisation is not over-selective on the graph layout, rendering representative visuals of (almost) arbitrary structures. But even in the case that all relations of an ontology have to be taken into account for the visualisation, a small world graph may be the perfect representation.

- A vital property of well-designed visualisation methods is visual appeal. It is important in making an interface inviting for the user to explore and navigate within the visualisation with ease. Watching clusters falling apart or emerging from composed nodes fails to grow tiresome but gives a perfect overview of the communities in the graph.

- The visualisation technique offers an easy and intuitive method for panning and zooming. The user is able to interactively select an arbitrary aperture of the hierarchy to explore. By manually adjusting the zoom level the user can autonomously decide on the level of detail of the current neighbourhood.

- An important part of the visualisation are the interactive possibilities the user gets presented with. S/he can actively browse and navigate within the visualisation, expand and collapse groups of nodes (i.e. clusters), or choose any desired part of the graph to explore in detail. This interaction helps in understanding the structure and semantics of the graphical representation, thus promoting cognitive processing tasks. It supports the discovery process and facilitates goal oriented decision making processes, when specific knowledge has to be obtained from the visualisation.

2.4 Summary

In this chapter we laid out the foundations of visual ontology alignment. The vision of the Semantic Web is to transform the loosely linked data repository, called the World Wide Web, into a global knowledge base by introducing semantic meaning to unstructured information. Ontologies are considered to be the back bone of the emerging Semantic Web phenomena. Their role is to archive a complete and consistent representation of a specific knowledge domain that can be queried and extended. The lack of an omnipotent universe ontology enforces us to deal with the alignment of competing ontologies to facilitate interaction of different applications.

The methodology of ontology mapping introduces knowledge sharing and reuse to the domain of ontology engineering by identifying the semantic differences between two ontologies. The task of ontology mapping is difficult and needs human supervision to generate accurate results. Concepts from information visualisation (InfoVis)

may provide leverage to the tedious process. However, the visualisation of a single ontology significantly differs from the task of visualising ontology alignment results. A promising graph rendering method to provide insight to both topics are clustering small world graphs. We believe, that the combination of various information visualisation techniques (e.g. linking and brushing, panning and zooming, detailed and global view, ...) can boost the performance of the overall task of visual ontology alignment.

Chapter 3

Visual Ontology Alignment

Even though encouraging results have been obtained in the last decade in the field of ontology integration, one major issue remains open: Accurate ontology mapping can only be achieved under the supervision of a human authority. The semantic ambiguity is too difficult for an automated process to resolve. Researchers are therefore putting more and more effort to assist human ontology engineers and provide helpful tools and methodologies to alleviate the overall process.

Information visualisation (InfoVis) proves to be a promising asset for this task. Interactive visualisation techniques help to shift the user task from largely cognitive into being more visual and physical, taking the work load off the human mind and storing intermediate information into an external representation [Ng, 2000]. Prerequisite is a well-designed and presented graphical representation that promotes information processing tasks and embeds the single steps within a uniform application.

In this chapter we discuss the formal methodology of visual ontology alignment and introduce AlViz [Lanzenberger and Sampson, 2006], a prototype application that implements a work flow according to the process definition introduced by Sampson [2007].

3.1 Task Description

The visual ontology alignment process consists of more than just a graphical representation. It has to be embedded in a consistent work flow including ontology selection, automated mapping computation, evaluation and correction, and re-iteration. The process of ontology alignment has been described by Ehrig and Sure [2004] and has been extended for visual ontology alignment by Sampson [2007].

We essentially agree with Sampson's proposal for a unified visual ontology alignment process, however, we would like to stress some details on the actual implementation. The visualisation is supposed to be a wrapper encasing the whole alignment process, thus providing a uniform front end to the alignment steps. The subtasks of mapping, evaluation, and correction (which comprise an iteration) have to be supervised by the user. This is done at best within and in accordance to the visualisation of the ontologies that are annotated according to the currently valid mapping. Before the first mapping the user is presented with a side-by-side view of the ontologies where s/he can mark obvious relationships to speed up the initial mapping process. Subsequently, the evaluation of the automated mapping is verified using the same visualisation, only augmented with the annotation of candidate mappings. If some corrections are necessary, the user is encouraged to point at the according node and adjust the relationship. After submitting the changes the next iteration can commence.

Following this informal task description the centre of the visual ontology alignment process is indeed the visualisation. But instead of placing it at the end of the process, it needs to be tailored as the central hub for all subsequent alignment steps. The subtasks are initiated from within the visualisation and return back to it after their successful completion to trigger adjustments to the overall representation of the ontologies and their alignments.

According to this specification we argue that the task of visual ontology alignment aims at the exploitation of visualisation as a cognitive tool for ontology alignment problem solving. In order to obtain satisfactory results the problem description has to provide answers for a number of requirements:

- The goal is to provide the end user with a visual representation of proposed alignments to enable understanding, agreement, and validation of the mappings.
- By providing a visually appealing and self-explanatory visualisation, the end user does not need to be an ontology engineer to understand the complex semantics of the ontology, but has to apprehend the meaning of the aligned entities.
- Multiple views are obligatory to provide access to both global and detailed aspects of the aligned data. The combination of different types of graph layouts — that is, displaying the ontology from different perspectives — facilitates the user to comprehend and navigate large information space.
- Follow the Visual Information Seeking Mantra: "Overview first, zoom and filter, then details-on-demand [Shneiderman, 1996]."
- Mappings between two ontology entities can bear different semantic similarities. The visualisation needs to make the type of similarity explicit.

- The tool needs to allow the user to validate candidate alignments, either by accepting or rejecting the proposed correspondences; to correct invalid proposed alignments; and to provide support of determining missing alignments.
- The design and implementation of the interactive tool needs to consider time complexity of the layout algorithm to be low to maintain satisfactory user interaction.
- The focal point of the navigation is concentrated on the entities involved in a specific alignment mapping, not the entire information space. The tool needs to provide a way to highlight the candidate entities and their embedding in the local neighbourhood.
- Provide structural context of the candidate alignments, so the user can explore the neighbourhood of the corresponding concepts.
- The layout of the visualisations needs to make efficient use of the available screen estate. The visual scalability is important to display a large number of concepts simultaneously, without presenting the user with a cluttered view.

After laying out the task requirements, in the following section we present the prototype version of our tool, AlViz, that aims to approach the problem of visual ontology alignment.

3.2 AlViz

AlViz has been designed and developed at the Institute of Software Technology and Interactive Systems at the Technical University of Vienna. It aims to facilitate the task of ontology alignment by providing a methodology for visualisation and verification of alignment results. AlViz is supposed to assist the user in the process of ontology alignment, offering an easy-to-use interface and a set of functions to speed up the task of finding accurate alignment mappings. With the juxtaposition of automated alignment mappings and the semantic capabilities of a human ontology engineer the performance and accuracy of the ontology alignment process can be increased. This ambition is pursued by exploitation of information visualisation techniques to present the user with a multi–view environment that enables visual comprehension of ontologies and their mapping results.

AlViz is founded on three cornerstones, which shall be discussed in this chapter: (1) Protégé (Section 3.2.1) is an ontology authoring tool, providing a framework for ontology editing and analysis. Designing AlViz to be a Protégé plug-in allows the exploitation of a well-defined and thoroughly tested application interface for ontology manipulation. (2) The initial alignment mappings are provided by the Framework for Ontology Alignment and Mapping (FOAM, Section 3.2.2), a tool for

(semi-)automated ontology mapping. FOAM has been designed to derive complex alignments based on similarity heuristics. (3) The ontology visualisation algorithm is based on Stephen Ingram's Small World Graph implementation (Section 3.2.3). Being a cluster visualisation, the interactive graph allows intuitive navigation and exploration of the graphical representation and allows the user to smoothly switch the level of detail.

The combination of these three technologies is the starting point for the implementation of a powerful tool for visual ontology alignment that offers a consistent work flow. AlViz assists the user during the complete process to efficiently and accurately verify and correct ontology alignment mappings. The work flow includes the selection of appropriate ontologies, the loading of preliminary mapping results, visualisation and exploration of ontology alignments, correction of misaligned mappings and storage of the final alignments. We believe that the current prototype version of AlViz significantly increases the speed and accuracy of ontology alignment. Future implementations will further reduce the tedious work load and improve the overall experience, leading to faster and more reliable results.

3.2.1 Protégé

Protégé[1] is a free open source ontology editor and knowledge base framework, which has been developed at the Stanford University and is supported by a large community of independent developers. Protégé has been chosen for its wide acceptance in the Semantic Web community and for its sophisticated support for manipulating OWL documents. Its extensible architecture allows the implementation of a wide range of add-ons and plug-ins. Being written in Java it runs on almost any platform and provides a full–fledged and mature Application Program Interface (API) for parsing, processing, and the visualisation of OWL documents.

The architecture of Protégé allows and invites the implementation of powerful extensions [Gennari et al., 2003] to the core application. There are few restrictions on the size and impact of these extensions, varying from small add-ons to (almost) stand–alone applications within a separate workspace — dubbed *tab-widget*.

AlViz has been developed as a tab-widget for Protégé, which brings the benefit of having an independent application whilst enjoying all features of a well-implemented API and the convenience of a consistent Graphical User Interface (GUI). The methods for manipulating OWL data structures are inherited from the Protégé API, whereas the visualisation and ontology alignment methods have been implemented autonomously. In this way the core application can easily be transferred to a similar framework with minimal effort by adapting the methods for GUI and OWL manipulation.

[1] http://protege.stanford.edu, (30.10.2009)

3.2.2 FOAM

In Section 2.2.3.2 we defined ontology mapping as the task of identifying similar semantic nodes in two ontologies. As we now approach tools for automated ontology alignment, we will need a formal — and therefore algorithmic expressible — definition of *similarity*. In the scope of this work we will refer to similarity as defined in Ehrig [2005]. Accordingly, the similarity function maps a pair of entities to a real number between 0 and 1, expressing the similarity between them. A calculated value of 1 means that the compared entities are identical, whereas a similarity of 0 implies that they are different and lack any common characteristics. This definition can be extended such that not only simple entities can be compared but also sets of composed entities (e.g. sub-ontologies).

The Framework for Ontology Alignment and Mapping[2] (FOAM) is a tool to fully or semi-automatically align two or more OWL ontologies based on similarity heuristics. It has been developed by Marc Ehrig and York Sure at the University of Karlsruhe. The goal of FOAM is to provide the user with a tool to return high quality alignment results (with explanations) for two given ontologies. This goal is pursued by six steps [Ehrig and Sure, 2005]:

1. **Feature Engineering:** As Ehrig [2005] states, "Entities are the same, if their features are similar", feature selection and composition are vital ingredients for a successful mapping approach. Thus, the primary step Feature Engineering selects excerpts of the overall ontology definition to describe a specific entity. This step is based on an educated guess about significant features (e.g. labels, subsumption, restrictions) which are then used for the actual comparison process.

2. **Search Step Selection:** In order to perform reasonably on large ontologies, it may be inefficient to compare each entity from one ontology with each entity of the other. Clever limitation of the comparing node-pairs can improve performance and accuracy. The Search Step Selection aims to reduce unnecessary comparison of distinct nodes by choosing promising pairs of nodes to be compared.

3. **Similarity Assessment:** This step indicates the actual similarity value for a given pair of candidate entities. Notice that the value also indicates equality (i.e. a value of 1) and diversity (i.e. a value of 0).

4. **Similarity Aggregation:** Based on the feature selection for any given pair of compared entities several similarity values are computed. These different similarity assessments have to be aggregated into a single measure.

[2] http://www.aifb.uni-karlsruhe.de/WBS/meh/foam/, (30.10.2009)

5. **Interpretation:** The interpretation strategy uses individual or aggregated similarity values to derive and propose an alignment. This is usually achieved with thresholds or combination of structural and similarity criteria. This step may also include user validation if a semi-automated approach is preferred.

6. **Iteration:** Identification of one entity pair influences the similarity of neighbouring entities. Thus, the similarity is propagated through the ontologies, forcing a re-calculation of neighbouring entity pairs. This step ends, when no new alignments are proposed.

After a predefined number of iterations the process returns a representation of alignment results, indicating the mapping relations between two given ontologies. The output of the results, however, is given as a long list of one-to-one accordances, representing the unique identification of the compared nodes and additional numeric values of similarities. The type of relation and the location of the nodes within the ontologies are not intuitively appreciated. The ontologies have to be physically examined to put the alignment results in place and validate the findings.

3.2.3 Small World Graph Visualisation

The graph visualisation algorithm used in AlViz is based on an implementation of Stephen Ingram [Ingram, 2005] incorporating the energy–based clustering LinLog model of Noack [2005] and a series of interactive techniques for cluster graph visualisation devised by van Ham and van Wijk [2004]. Ingram's Java implementation — dubbed Small World Graph Visualisation — exploits prefuse[3] as visualisation toolkit to develop an interactive graphical representation, and Colt[4], a set of Open Source Libraries for High Performance Scientific and Technical Computing.

We implement both of these algorithms using an efficient sparse matrix formulation. In practice we use the sparse matrix facilities provided by Colt, a set of Open Source Libraries for High Perfor- mance Scientific and Technical Computing in Java.

The Small World Graph Visualisation is a clustering algorithm, establishing relations between nearby nodes and visualising neighbourhoods of close entities. The cluster visualisation helps a user to examine the structure of the ontology intuitively. The level of detail can be adjusted by a zoom level slider on the right hand side of the screen, thus enabling the user to explore the cluster hierarchy gradually.

Similarity is derived from the weights between two nodes. Edges between similar nodes are shorter, placing the nodes closer to each other. The clustering algorithm decides which nodes become merged into a single cluster. Navigation within the graph is established by panning and zooming. The global view shows individual

[3] http://www.prefuse.org/, (30.10.2009)
[4] http://acs.lbl.gov/~hoschek/colt/, (30.10.2009)

nodes that are placed within the graph according to their relations. Zooming out leads to the merging of nearby nodes to a larger cluster. By seamless zooming and panning every part of the graph can be examined in detail. The animated composition or decomposition of clusters conveys the graph structure to the user.

The algorithm takes a tuple of two nodes and an optional value of weighting as input parameters. In the current implementation of AlViz, the inheritance relation is used to render the graph layout: The nodes represent ontology classes and the vertices represent the *is-a* relations. However, in future releases the user will be provided with a selection of available (role-)relation of the ontology to chose from.

The weighting of the links has a significant impact on the graph layout as it is used to compute the distance between two nodes (i.e. the edge), thus inducing a level of neighbourhood. Adjacent nodes with a higher weighting value are placed closer to each other than nodes with a smaller value. Therefore the weighting primarily determines whether two nodes are part of the same cluster.

In the current setup the graph nodes are visualised based on the *is-a* relation of the ontology, ignoring transitive relations. Thus only nodes that are a direct subclass of their predecessors are taken into account. Such relations are treated equally, represented with the same weighting factor. However, the configuration can easily be adjusted to represent transitive relations by applying different weightings depending on the *is-a*–distance between two nodes.

3.2.4 Putting it all together

Protégé is the underlying framework, providing the methods for ontology manipulation on the data layer. FOAM is used to deliver preliminary mapping results for the initial alignment. Finally, the clustering algorithms used for the visualisation are Small-World Graphs.

Putting it all together, the work flow starts with an automated mapping algorithm applied on the given ontologies. The results are loaded into the AlViz plug-in to prepare a first draft of intended mappings. These preliminary mapping results are incorporated and visualised. The user is then presented with a coloured and linked graph where the colours represent semantic proximity between the linked entities of the given ontologies. The classes found to have correspondences are marked and linked with their counterpart in the opposed ontology. Finally, the eligible classes can be examined in detail by exploring their properties and neighbourhood based on their ontology definition. AlViz incorporates multiple techniques to deliver and present complete information in such a way that a human ontology engineer can finalise the alignment process with minimal effort.

The core feature of AlViz is the excessive use of information visualisation techniques, presenting the user with various options and possibilities to explore and analyse both ontologies and the relationships between them. The use of multi-

ple views showing different interpretations of the same information augments the understanding of the structure and supports consistent navigation within the ontologies.

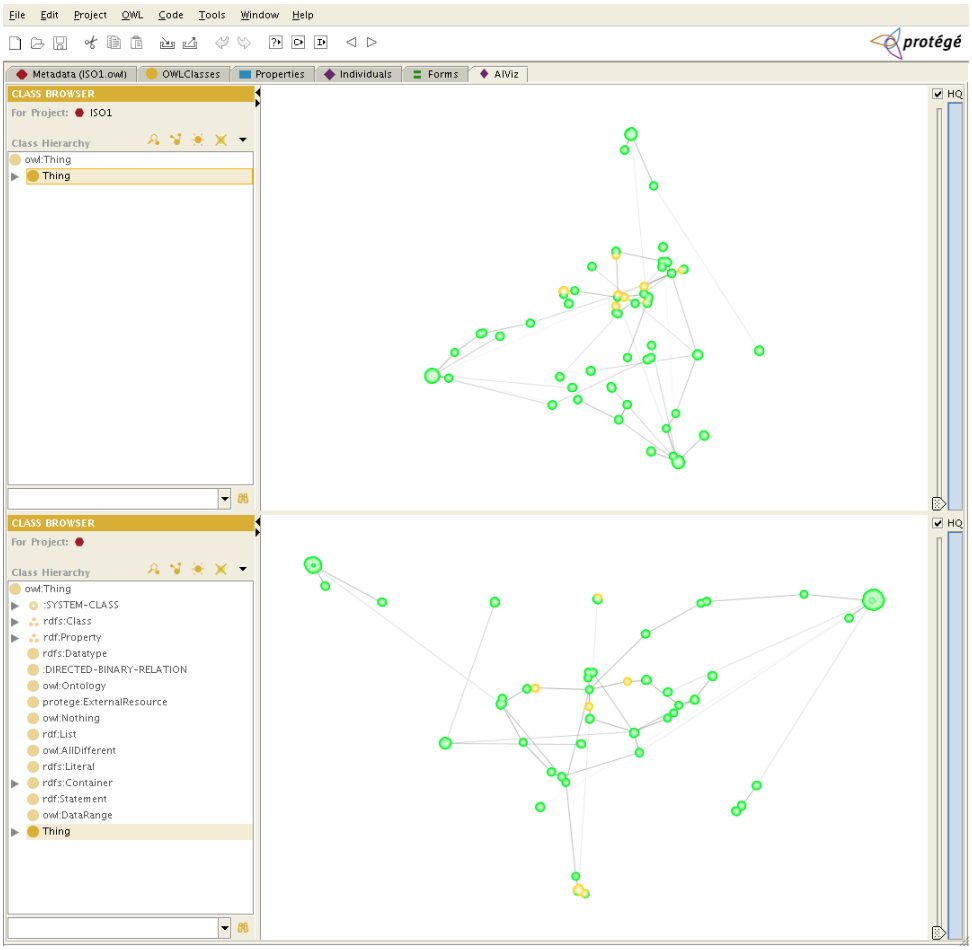

Figure 3.1: The initial startup screen of AlViz showing an overview of the application.

The main screen of AlViz is divided in two identical parts placed on top of each other (Figure 3.1). The first ontology is placed in the upper half of the screen, while the second resides in the lower half. Each ontology has a separate view plane consisting of two graphical interpretations.

The presented view of an ontology[5] consists of three separate visualisation techniques. First, the graph visualisation is computed based on the *is-a* relation of the ontology classes. Every class is treated as a node, whereas an *is-a* relation between two classes represents an edge. Based on the edge count, the distance between two nodes is determined, placing entities sharing many edges closer to each other. The graph constitutes a cluster visualisation. By zooming out — in manually shifting the slider on the right hand side of the graph — groups of nodes collapse, emerging into a single but larger node. Pulling the slider in the opposite direction increases the detail level again, making the clusters to fall apart and reveal the individual nodes.

The second technique handles the automated alignment results provided by an external mapping tool — In this early stage of development only one mapping algorithm (i.e. FOAM) is available. Future implementations will be equipped with a selection menu for the user to choose from a list of different ontology mapping tools. The classes identified to have relations to classes in the opposing ontology are colourised according to their particular semantic relation. Each relation category is associated with a specific colour, for a user to easily perceive the semantics of the alignments. The colourisation is placed on top of the cluster visualisation, representing a merged screen of detailed and alignment view, showing both the source ontologies and the alignments that hold between them. In conjunction with the nested cluster visualisation the clusters change colours to show the predominant alignment types in a branch of the ontology at different zoom levels.

Finally, for the third technique both ontologies are linked with a view of the J-Tree class browser visualisation, showing the taxonomy of the appropriate ontology. The presented view (Figure 3.1) enables the user to explore each ontology by multiple means and in great detail. Due to the clustering algorithm users can pan and zoom into every area of the graph, enabling them to select a region of interest and explore the graph in detail. The graph and the J-Tree representations are internally linked, such that changing the focus in one visualisation induces a change in focus in the other. That is, by moving the mouse to a specific node in the graph and pressing the mouse button the node is marked as *selected*. This selection is linked back to the J-Tree where the corresponding node is highlighted and selected as well. In case of a (partly) collapsed J-Tree, the thread leading to the appropriate node is opened automatically, rendering the class visible. This technique is called *linking and brushing* and works in both ways: It can be initialised from either the J-Tree or the small world graph. Figure 3.2 shows an example of this technique.

These visualisation techniques are used for both ontologies. So each half of the screen is treated equally and is provided with the same methods for analysing, exploring, and examining the according ontology. But the ontology panels are not

[5] The visualisation techniques used for both ontologies are identical and differ only on the specific calculations based on the underlying ontology. The technical details on the user interface and visualisation methods are valid for both instances.

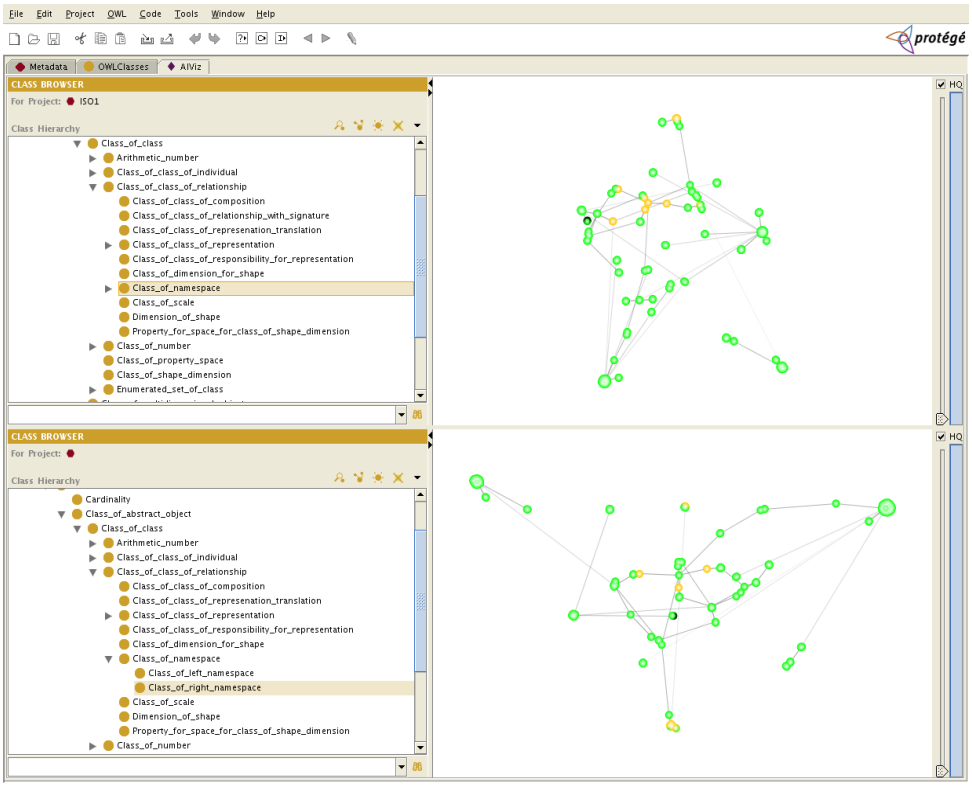

Figure 3.2: The nodes annotated with a black shadow are identified to be similar in both ontologies. Linking and brushing connects the appropriate nodes in all four ontology representations and highlights their location.

entirely independent. One vital methodology has been implemented for the visual comparison of the observed ontologies. Linking and brushing is propagated through both ontologies. That is, selecting a class in either J-Tree or small world visualisation in one ontology highlights the associated class in the opposed ontology, if a semantic relation has been specified that holds between them. In the case that a node is selected that fails to have a corresponding entity, any previous established selection in the opposed ontology gets revoked, dissolving the highlighting.

The main features of AlViz are indisputably the various visualisation techniques that allow users to quickly gain an overview of the ontologies at hand and visually explore the mappings between them. However, for a satisfying user experience the work flow has to be optimised and convenient methods implemented to handle the loading and storage of ontologies and their corresponding alignment results. These improvements will be a main concern in future versions. The current prototype acts as a proof of concept for visual ontology alignment.

The following sections describe the technical details and implementation issues that occurred during the development.

3.3 Technical Design

The technical development of AlViz is still a work in progress. The version described in this thesis is an early prototype that is already mature enough to prove its usability and its suitability to assist a human user in the difficult task of ontology alignment.

AlViz has been developed with a modularised approach in mind. The technical implementation resembles the work flow of the overall task. Every step of the work flow has been implemented in a separate Java class to allow for independent development of different parts in the application, and to enable the replacement of individual steps. The application — and therefore the work flow — is divided into five major steps:

Initial Alignment Mapping. AlViz is dependent on external tools to provide alignment mappings. For a smooth and fast delivery we suggest the use of (semi-)automatic ontology mapping algorithms such as FOAM.

Transformation of Alignment Results. External tools usually deliver their results in their specific representation. This step coordinates the transformation of those tool-dependent mappings into the general AlViz alignment format.

Loading and Parsing. The first task of the work flow involves the loading of the ontologies and their corresponding alignment mappings. The parsing of those external resources is explained in this section.

The visual representations of class browser and graph visualisation is the core functionality of AlViz. The characteristics of both visualisation techniques and their technical coupling is of eminent importance to the fundamental understanding of AlViz.

The AlViz Tab Widget. This section describes the global perspective: The main classes and methods of the implementation, how they interoperate, and how they compose the final application.

Every step has well-defined prerequisites and a specific outcome that is required for the subsequent task. We shall discuss every step in detail.

3.3.1 Initial Alignment Mapping

AlViz does not implement automated alignment methods. It depends on external tools to provide the initial mappings. This design decision allows the usage of various and multiple alignment algorithms, and they can even be combined to deliver more accurate results. As we have discussed in Section 2.2.6, there are no automated alignment tools that perform reasonable on arbitrary input data. There are, however, various algorithms that perform well on specific ontologies or on specific domains. The modularised structure of AlViz pays tribute to these findings and allows the usage of various mapping tools to increase the quality of ontology alignment. It remains to be the decision of the ontology engineer to evaluate promising ontology alignment tools to provide accurate initial mapping results.

However, the output listings generated by these tools can usually not be directly fed into AlViz. It has to be transformed into an generalised alignment result format. Unfortunately, the specific requirements of AlViz exceed the specifications of the Ontology Alignment API introduced by Euzenat [2004] so that the current implementation requires mapping results to be encoded according to the AlViz mapping format.

In this thesis we will concentrate on a set up in which FOAM has been used to provide initial alignment mappings. FOAM has various parameters that can be set using a separate file. In the basic setup five parameters are mandatory: The file locations of the two ontologies, the file location of pre-known alignments, the specification of a scenario (whether the mapping process is intended for query rewriting, ontology merging, data integration, reasoning, ontology evolution, or no predefined scenario at all), and the file location for alignments that get cut off during the mapping process. Additional, but optional parameters, include the maximum number of iterations, the use of a specific decision strategy, the cut-off value, whether to ask the user for input on doubtful findings (semi-automatic approach), the file location to store the results, and whether multiple alignments for each entity are allowed or removed.

In the default configuration of FOAM the algorithm yields a binary result for any two nodes it compares, identifying identical or distinct classes. However, the underlying mapping process does distinguish various level of similarities so that in "collaboration with the developer of Foam we found correspondences between our categories and 23 rules in the alignment algorithm." [Sampson, 2007] The outcome of this collaboration is a specific configuration in which the FOAM algorithm delivers a complete list of 23 assets, each describing a specific similarity class and the value at which the relation between the two nodes adhere to this characterisation. A complete listing of identified similarity classes can be found in Table A-1 in the appendix. This is necessary because AlViz handles various similarity classes that are visualised accordingly. For a domain expert or ontology engineer with the task of aligning two different ontologies, it is essential to distinguish between close

similarities and obvious disparities. AlViz renders such properties with different colourisation, allowing the end user to confirm or reject the mappings.

3.3.2 Transformation of Alignment Results

The default output format of FOAM representing the mapping results yields in its basic form a list of semicolon–separated Universal Resource Identifiers (URI), indicating, that the two nodes are identical. Depending on the chosen algorithm options, a variety of additional fields are computed, representing for example, the type of alignment similarity, a confidence factor, or the correctness of the match. All of these additional values are appended to the initial tuple of URIs and separated by semicolons. In the specific configuration that feeds the visualisation algorithm of AlViz, the final alignment results are delivered in a flat file. Unfortunately, such a representation tends to be cluttered and confusing, which makes them difficult to comprehend without some introduction or documentation. A short extract of an example output file has been provided in Listing 3.1, showing the results of the alignment mapping between the ontology *animalA* and the ontology *animalB*.

Listing 3.1: FOAM output file

```
http://www.atl.lmco.com/projects/ontology/ontologies/animals/animalsA.owl#
    TwoLeggedPerson; null; http://www.atl.lmco.com/projects/ontology/ontologies/
    animals/animalsB.owl#Hermaphrodite; null; 0.10083160083160081; -0.3; -1.0; -1.0;
    -1.0; -0.3; -1.0; 0.0; 0.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; 0.0; -1.0; -1.0;
    -1.0; -1.0; -1.0; 0.0; 0.0; -1.0; -1.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; -1.0;
    0.0; 0.0; -1.0; -1.0; -1.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0;
    0.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0;
    -1.0; -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -0.3; -1.0; -1.0; -1.0; -1.0; -1.0;
    -1.0; -1.0; -1.0; -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -0.3;
    -1.0; -1.0; -1.0; -0.3; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -0.3;
    -1.0; -0.3; -1.0; -1.0; -0.3; -0.3; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0;
    1.0; 0
http://www.atl.lmco.com/projects/ontology/ontologies/animals/animalsA.owl#
    TwoLeggedPerson; null; http://www.atl.lmco.com/projects/ontology/ontologies/
    animals/animalsB.owl#Animal; Animal; 0.10083160083160081; -0.1; -1.0; -1.0; -1.0;
     -0.3; -1.0; 0.0; 0.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; 0.0; -1.0; -1.0; -1.0;
     -1.0; -1.0; 0.0; 0.0; -1.0; -1.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; -1.0; 0.0;
    0.0; -1.0; -1.0; -1.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; 0.0;
    -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0;
    -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -0.3; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0;
    -1.0; -1.0; -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -0.3; -1.0;
    -1.0; -1.0; -0.3; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -0.3; -1.0;
    -0.3; -1.0; -1.0; -0.3; -0.3; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; 1.0; 0
http://www.atl.lmco.com/projects/ontology/ontologies/animals/animalsA.owl#
    TwoLeggedPerson; null; http://www.atl.lmco.com/projects/ontology/ontologies/
    animals/animalsB.owl#Woman; null; 0.10083160083160081; -0.3; -1.0; -1.0; -1.0;
    -0.3; -1.0; 0.0; 0.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; 0.0; -1.0; -1.0; -1.0;
    -1.0; -1.0; 0.0; 0.0; -1.0; -1.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; -1.0; 0.0;
    0.0; -1.0; -1.0; -1.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; 0.0;
    -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0;
    -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -0.3; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0;
    -1.0; -1.0; -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -0.3; -1.0;
    -1.0; -1.0; -0.3; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -0.3; -1.0;
    -0.3; -1.0; -1.0; -0.3; -0.3; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; 1.0; 0
http://www.atl.lmco.com/projects/ontology/ontologies/animals/animalsA.owl#
    TwoLeggedPerson; null; http://www.atl.lmco.com/projects/ontology/ontologies/
    animals/animalsB.owl#Man; null; 0.10083160083160081; -0.3; -1.0; -1.0; -1.0;
    -0.3; -1.0; 0.0; 0.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; 0.0; -1.0; -1.0; -1.0;
    -1.0; -1.0; 0.0; 0.0; -1.0; -1.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; -1.0; 0.0;
    0.0; -1.0; -1.0; -1.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; 0.0;
```

```
    -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0;
    -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -0.3; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0;
    -1.0; -1.0; -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -0.3; -1.0;
    -1.0; -1.0; -0.3; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -0.3; -1.0;
    -0.3; -1.0; -1.0; -0.3; -0.3; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; 1.0; 0
```

AlViz is unable to use the default output format of FOAM directly. Rather, the alignment mapping has to be transformed into a well-defined XML format that can be interpreted by the AlViz parser. We have documented the specification of the XML format in both XML Schema[6] (XSD) and Document Type Definition[7] (DTD). The specifications are listed in the Appendix (Listing A-1 and A-2).

The mapping results have to be dissected and each identified item specifically annotated. This transformation has been implemented by Jennifer Sampson with an Extensible Stylesheet Language Transformations[8](XSLT) processor to transform the initial flat file format into a well-formed XML Document that is validated against the AlViz mapping format specification.

Listing 3.2: AlViz input file

```
<?xml version="1.0" encoding="UTF-8"?>
<AlViz>
  <Nodes>
    <Node uri="http://www.atl.lmco.com/projects/ontology/ontologies/animals/animalsA.
        owl#TwoLeggedPerson">
      <Entity label="null" type="" />
        <CNode uri="http://www.atl.lmco.com/projects/ontology/ontologies/animals/
            animalsB.owl#Hermaphrodite">
          <Entity label="null" type="" />
          <SimilarityInfo Correctvalue="-1.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; -1.0;
              0.0; 0.0; -1.0; -1.0; -1.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; -1.0;
              -1.0; 0.0; 0.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0;
              -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -0.3;
              -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -0.3;
              -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -0.3; -1.0; -1.0; -1.0; -0.3;
              -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -0.3; -1.0; -0.3;
              -1.0; -1.0; -0.3; -0.3; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; 1.0;
              0">
          <Confidence value="0.10083160083160081" />
          </SimilarityInfo>
        </CNode>
    </Node>
    <Node uri="http://www.atl.lmco.com/projects/ontology/ontologies/animals/animalsA.
        owl#TwoLeggedPerson">
      <Entity label="null" type="" />
        <CNode uri="http://www.atl.lmco.com/projects/ontology/ontologies/animals/
            animalsB.owl#Animal">
          <Entity label="Animal" type="" />
          <SimilarityInfo Correctvalue="-1.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; -1.0;
              0.0; 0.0; -1.0; -1.0; -1.0; -1.0; -1.0; 0.0; -1.0; -1.0; 0.0; -1.0;
              -1.0; 0.0; 0.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0;
              -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -0.3;
              -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -0.3;
              -1.0; -1.0; -0.3; -1.0; -1.0; -0.3; -0.3; -1.0; -1.0; -1.0; -0.3;
              -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -0.3; -1.0; -0.3;
              -1.0; -1.0; -0.3; -0.3; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; -1.0; 1.0;
              0">
          <Confidence value="0.10083160083160081" />
          </SimilarityInfo>
```

[6] http://www.w3.org/TR/xmlschema-0/, (30.10.2009)

[7] http://www.w3.org/TR/REC-xml/#dt-doctype, (30.10.2009)

[8] http://www.w3.org/TR/xslt, (30.10.2009)

```
            </CNode>
        </Node>
    </Nodes>
</AlViz>
```

After the transformation into the XML format (Listing 3.2) a human user can easily perceive its semantics. It may still be overloaded and difficult to capture all individual entries, but for a quick look or verification of single results it is more than sufficient. But the XML format brings advantages for automated parsing as well. The hierarchical structure allows the parser to navigate within the document, selecting and extracting only relevant parts. Future extension of the format can be written in such a way that parsers, relying on the old format, do not fail to identify the data that it is still able to process.

The preparation of the alignment mappings is currently an external task. Obviously, it is necessary to embed such a vital process into the core application as soon as possible. However, the functionality will remain the same: (1) Configuration of the mapping algorithm, (2) initiate the mapping process, (3) retrieve the results and (4) transform them into the appropriate XML format. These steps will be included in future versions of AlViz.

3.3.3 Loading and Parsing

Three components have to be loaded to initialise the alignment framework. First, the two ontologies to be aligned are loaded either from a prepared Protégé project or by manual selection. Being a Protégé plug-in AlViz uses Protégé built-in methods and data structures to load, store and represent ontologies. Thus, the first ontology can be extracted and loaded from any Protégé project. The second ontology has to be imported from a valid OWL file.

The third component to be loaded is a list of preliminary alignment results, used for an initial alignment visualisation. Unfortunately, Protégé does not provide developers with an standardised method to retrieve and store alignment results. Nevertheless, it is possible to store these results within the Protégé project file, which shall be implemented in a future version of AlViz. Currently, the alignment results have to be loaded and stored manually from a separate XML file.

During the parsing process of the alignment results, the semantic relations are identified and preserved for subsequent exploitation. For the final visualisation these relations will be used to mark corresponding nodes. Each relation is associated with a specific colour, thus graph nodes that hold a specific relation are coloured accordingly.

FOAM actually identifies 23 different semantic relations, which are encoded in the alignment results file. Whereas all of these relations are valid and useful, some of them imply similar semantic findings. Moreover, presenting the user with a clut-

tered graph containing multiple similar colours may lead to distraction and even aggravate perception and cognition of the information visualisation. Treisman [1985] explains in her theory of preattentive processing that specific visual features do not necessarily require to be processed by the complete visual and cerebral system but can be rapidly and accurately be identified by our low–level visual system.

> Preattentive processing of visual information is performed automatically on the entire visual field detecting basic features of objects in the display. Such basic features include colors, closure, line ends, contrast, tilt, curvature and size. These simple features are extracted from the visual display in the preattentive system and later joined in the focused attention system into coherent objects. Preattentive processing is done quickly, effortlessly and in parallel without any attention being focused on the display. [Treisman, 1985]

Of course, this *short cut* has its limits. If there are more than seven colours used for a visualisation, the preattentive processing will not be applied by our brain. Instead, sequential scanning would be required for a successful perception, resulting in a much longer process.

Therefore, to limit the amount of colour codes some of the available semantic relations are subsumed and represented by the same colour. The implemented compilation of semantic relation and the corresponding colour code is listed in Table 3.1. The full list and the corresponding colour code can be examined in Table A-1 from the appendix.

semantic relation	**colour**
equal	red
syntactically equal	orange
similar	green
broader	blue
narrower	magenta
different	yellow

Table 3.1: Colour code

The XML encoded alignment results are taken by the AlViz parser and validated against the AlViz XSD. For reading and validating XML structures, the SAX parser[9] has been utilised. If a well-formed and successfully validated XML structure is identified, a nested *Hashtable* data structure is constructed, representing the

[9] http://www.saxproject.org/, (30.10.2009)

alignment results. The generated data structure resembles a look-up table to facilitate mapping between linked nodes. This feature is excessively exploited in the interactive visualisation for linking and brushing.

The FOAM algorithm tries to find all relations that hold between two entities. If a comparison of two nodes yields the result that they are *equal* then they subsequently tend to be *syntactically equal* and *similar* as well. AlViz is only interested in the strongest semantic relation (i.e. in this case *equal*). However, the alignment result format allows the specification of more than one semantic relation to be specified for any two nodes. These features are also represented in the *Hashtable*, but, for easier access, the strongest similarity relation has been extracted and stored separately. The ranking of *strong* relations can be seen in Table 3.1, reading from top to bottom.

All subsequent computation of the ontologies is based on the *Hashtable* data structure. The input files are closed and remain untouched.

3.3.4 The Visual Representations

The global view of AlViz consists of four individual visualisations, two for each ontology. The view pane is thus divided twice, showing two representations of the first ontology in the upper half of the screen and two for the second ontology below. The technical implementation is based on a nested *GridLayout* architecture, where each ontology is referenced by a *JPanel* to adjust the visualisations independently. Both *JPanels* are then integrated within an additional *GridLayout* constituting the global view.

For the remainder of this thesis we will refer to the upper half of the screen as *first ontology panel* and the lower half as *second ontology panel*. Each ontology panel consists of a J-Tree representation and a small world representation of the ontology. The ontology panels are identical with the exception that their visual representations are each based on the respective ontology.

The J-Tree representation is an inherent functionality of Protégé and needed only few modifications to suit our needs. In order to enable linking and brushing a new Java class has been implemented that extends the Protégé class accordingly. The added functionality includes the tracking of selected entities, the highlighting of appropriate nodes, and the methods for collapsing/expanding a sub tree if the selected node is currently collapsed.

Severe modification had to be implemented to Ingram's source code to accommodate the design of AlViz: The algorithm places the graph randomly on the screen, presenting the user with different topologies of the same ontology every time it is loaded. The AlViz implementation has been modified in order to disable the random initial placement of nodes. This adaptation has been introduced to present the user with a consistent view of an ontology. For a complete alignment process it may

be necessary to run multiple sessions, for why the user must be supported to easily recognise already known ontologies and continue the work where s/he left.

Furthermore, the original code does not support independent colourisation of single nodes or clusters. This extension had to be implemented in order to emphasise similarity results obtained by the automated ontology mapping. The colour of a single node is determined by the initial mapping file according the mapping in Listing A-1. The *strongest* relation determines the specific colour of a node. Beyond that, if adjacent nodes get visually merged into a cluster when zooming out, the colour of the appearing cluster node is determined by majority vote. If more than 50 percent of the nodes share a common colour, this colour is propagated and used for the emerging cluster node. This behaviour may change in future versions. An alternative implementation where the colour of the strongest relationship is chosen for the emerging cluster has been discussed, but temporary suspended until an according user study has been conducted.

Again, additional methods have been implemented to enable linking and brushing. The user interacts with the visualisation using the mouse pointer as a focal tool. In a lower zoom level the focal tool acts as a geometric fish eye distortion within the operating diameter of the virtual fish eye lens. The focused clusters collapse and reveal the single nodes that constitute the specific cluster. Ingram's version draws concentric circles around the operating diameter of the lens and unfolds the identifier of the focused node in a tool tip. The user actively moves the focus tool and gets immediate response of its sphere of action. The indication of the focus diameter is thus redundant and has been removed. Similarly, identification of the focused node is only relevant when the node is selected. Using the linking and brushing technique, the according node is selected in the adjacent J-Tree where the node name is displayed prominently. The tool tip is therefore unnecessary and has been secluded.

Some modifications were necessary to enforce the use of specific parameters that are optional in the original implementation. Ingram provides us with two clustering techniques. The first one to be *Average Link*, where two neighbouring clusters are merged if their average shared edge length is the smallest. The second implementation of a clustering algorithm by Newman [2004] was specifically designed to detect communities in small world graphs. With the current implementation of AlViz the Average Link algorithm is preferred because the clustering algorithm pays tribute to the geometrical properties of the graph, allowing neighbouring clusters of a community to emerge simultaneously. Moreover, the algorithm generates an even-distributed topology, expanding the graph in all directions equally. The Newman algorithm, on the other side, tends to stretch out the node distribution along the horizontal axis, aligning the clusters equally on the available screen estate. For small ontologies this method actually minimises cluster accumulation. We believe that this property may even hinder the interactive visualisation because the user is presented with an almost static graph. Thus, in the current implementation the op-

tion to invoke the Newman algorithm has been removed. However, following the design principles of AlViz, we intend to include various different graph rendering algorithms in future versions. The user will then be able to choose a visualisation method that suits best to the given ontologies.

The Small World Graph visualisation provides three different visualisation methods for edge rendering: line edges, tube edges, or no edges at all. For AlViz this selection has been withdrawn, enforcing the rendering of simple line edges to explicitly indicate the relations among the nodes. No edges would refute these relations, whereas the drawing of thick tube edges would cause distraction to the cluster topology, drawing too much attention off the clusters. However, the evaluation study conducted by Huber [2009] revealed that users noticed that the edge thickness does not reveal any relevance in the relation between two nodes. This indicates that it may be necessary to adjust the edge thickness according to the distance.

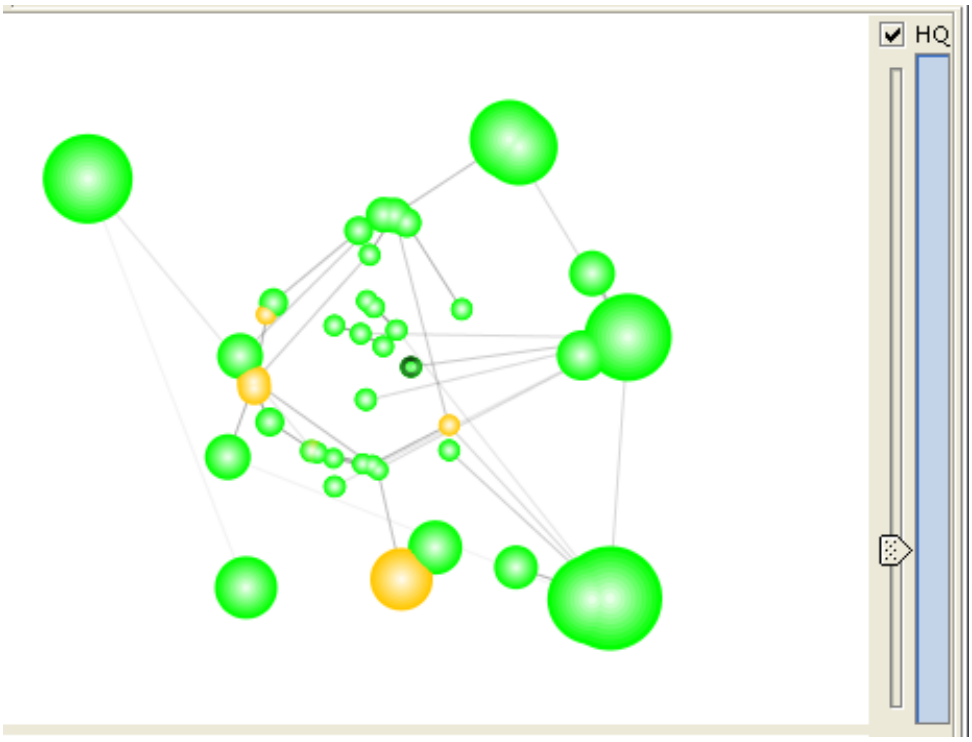

Figure 3.3: The small world graph representation used in AlViz showing the cluster visualisation of an ontology.

Finally, some visual properties (e.g. slider controls and quality settings) have been adjusted to better fit the conceptional layout of AlViz. The finalized version of the small world graph can be observed in Figure 3.3. For a comparison to the original layout please refer to Figure 2.3 on page 35.

The rendering of the small world graph is implemented using the prefuse visualisation toolkit, a Java library that supports a rich set of features for data modelling, visualisation, and interaction. The technical implementation has been adopted from Ingram's version. The corresponding Java library got incorporated into the AlViz package and is distributed with the plug-in. The rendering of the class browser is a built-in method of Protégé and did not needed to be modified.

3.3.5 The AlViz Tab Widget

As has been stated earlier, the AlViz application is embedded into the Protégé framework as a tab widget. This plug-in method allows the application to be reasonable independent but to exploit vital functions of the powerful Protégé API. In order to enable this functionality, the AlViz main class has to extend the *AbstractTabWidget* class provided by Protégé and implement an appropriate *initialize()* method, where the main panel and its layout is specified. The logic to fill and maintain the content of the tab widget is defined in the *SmallWorldParent* method. This is the place where the file dialogue gets initiated, the ontology panels are constructed, and the overall layout of the application is determined.

A single ontology panel is described in the *SmallWorldFrame* class. It is composed from the extended class browser *AlVizClsesPanel* object, which constitutes the extended class browser class for the J-Tree, and the *FoamData* object, which puts together the ontology structure and the alignment results provided by FOAM.

The *FoamData* class is actually just a meta container providing methods for addressing and retrieving ontology data. The parsing of XML encoded alignment results is performed in the *FoamReader* class. It implements the control structures for reading and analysing XML data provided by the SAX parser and compiles a data structure representing the mapping results. This data structure is then combined with the data resembling the corresponding ontology. The information concerning the ontology configuration is retrieved using Protégé methods for accessing OWL based knowledge bases. The consolidation of mappings and ontology entities is realised within the *AlVizReader* class. This process identifies the similarity rules and categorises them according to the specifications from Listing A-1.

This final step concludes the generation of the *Hashtable* that represents the ontology. This data structure is vital to the utilisation and performance of AlViz, as all ontology and graph operation methods are propagated and interpreted solely through the manipulation of the *Hashtable*.

3.4 Summary

In this chapter we described the novel approach of visual ontology alignment [Lanzenberger and Sampson, 2006] and presented a prototype application that incorporates the findings. AlViz embeds the requirements of visual ontology alignment [Sampson, 2007] into an uniform application that aims to guides the user through the complete work flow and assists by providing an easy-to-use interface. The tool is implemented as multi–view plug-in for Protégé using J-Trees and small world graphs. A total number of four visualisations are provided to allow the user to freely explore the ontologies in detail. The user is provided with various visualisation techniques to interact with the representations: magnifying, change of focus, colourisation, panning and zooming, and linking and brushing.

In this early version of AlViz the focus of research was to confirm the requirements of visual ontology alignment and to evaluate an application design based on these principles. We recognize that the tool is still missing some vital features and enhancements to fully integrate the necessary work flow. However, the current version supports our argument, that visual ontology alignment is a promising conceptual approach to master the difficult task of ontology alignment.

Chapter 4

Evaluation and Outlook

In this chapter we describe all necessary steps to complete a validation process of externally prepared alignment mappings. By means of a use case scenario we illustrate the visualisation capabilities of AlViz. We deem this approach as a perfect fit to introduce all core features of AlViz to point out and explain the benefits of a multi-view visualisation environment.

The Section 4.3 provides an analysis of the results of the survey. They are recapped, assessed, and compared to the initial requirements of the task of visually aligning ontologies. We evaluate in which respects AlViz meets the proposed requirements, and where it surpasses our initial goals.

In the final section we address those aspects where further development is necessary. We compile a comprehensive expose of important enhancements to fully address the given problem description of visual ontology alignment, and propose improvements to fulfil the requirements of usability and performance.

4.1 A Use Case

The most efficient way for the assessment of viability or even usefulness of a given tool is usually also the easiest: To use it. We therefore deem an illustrated use case scenario of a common validation work flow most appropriate to introduce the capabilities of AlViz.

This survey is conducted under the premise that the initial alignment mappings have been prepared by external means. In our case, the mapping tool FOAM has been used to provide the necessary mapping results. Also, the FOAM specific output file has been transformed into the XML based AlViz alignment format.

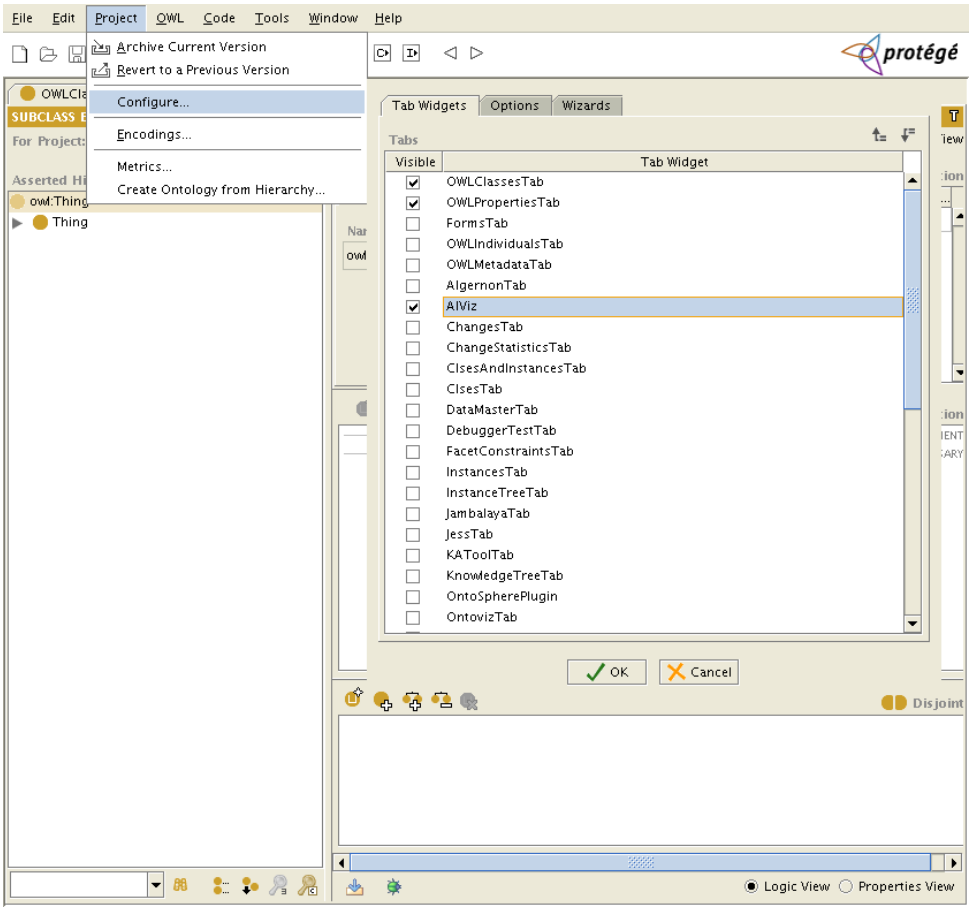

Figure 4.1: Activating the AlViz tab widget in the Protégé configuration menu.

First of all, the AlViz tab widget has to be activated to be used within Protégé. This can be done from the *Project*/*Configure* dialogue (Figure 4.1).

Right after the initialisation of AlViz two separate file dialogues pop up consecutively. The first one (Figure 4.2) asks for the location of the mapping results provided by the preceding FOAM mapping algorithm, whereas the second dialogue specifies the location of the second ontology.

After the selection of the corresponding files the application needs a few seconds to commence the visualisation. During this short start up period the ontologies are loaded, analysed, and consolidated with the mapping results. The progress of the final rendering is indicated with a vertical status bar on the far right side of the screen. When the blue bar reaches the top of the window the rendering is considered complete.

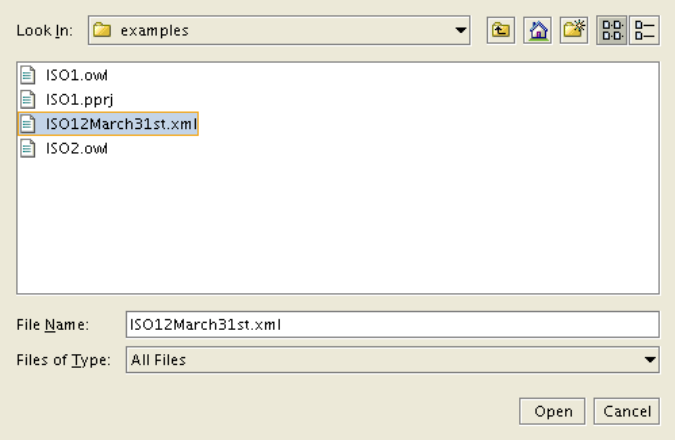

Figure 4.2: File dialogue for the location of external alignment results.

The start up screen of AlViz (Figure 4.3) displays four ontology representation — two for each ontology. The J-Trees on the left hand side are collapsed, whereas the small world graphs are initialised at an intermediate zoom level, giving an estimate overview of the topology of the visualised ontologies. The screen shot shows individually sized cluster nodes, all sharing the same green colour. According to the similarity class matrix (Listing A-1) they indicate that the majority of the nodes, emerging into the larger cluster, share a profound similarity to entities in the opposed ontology.

By using the mouse pointer as focal tool, the inherent configuration of the clusters can be examined without adjusting the overall zoom level. Figure 4.4 shows the effect of hovering the mouse pointer over the cluster in the lower right corner of the visualisation. The cluster vanishes and reveals the comprising nodes. This technique resembles an intensification of zoom level, bounded to the local neighbourhood in a predefined diameter of the mouse pointer.

The local examination also exposes some orange coloured nodes in the vicinity of the focused cluster. They indicate *syntactically equal* entities in the opposed ontology. Withdrawing the mouse pointer makes the nodes collapse again, re-merging the nodes into subsumed cluster.

To permanently expand the cluster configuration the zoom level can be adjusted to increase the granularity. This can be achieved by pulling the slider on the right hand side to the lowest setting. Figure 4.5 shows the graph in the highest zoom level. This view reveals an increased number of orange nodes that indicate a strong similarity between entities of both ontology.

The strength of AlViz lies in the combination of all four visual representations. Using the mouse pointer to select an individual node in the graph visualisation lays

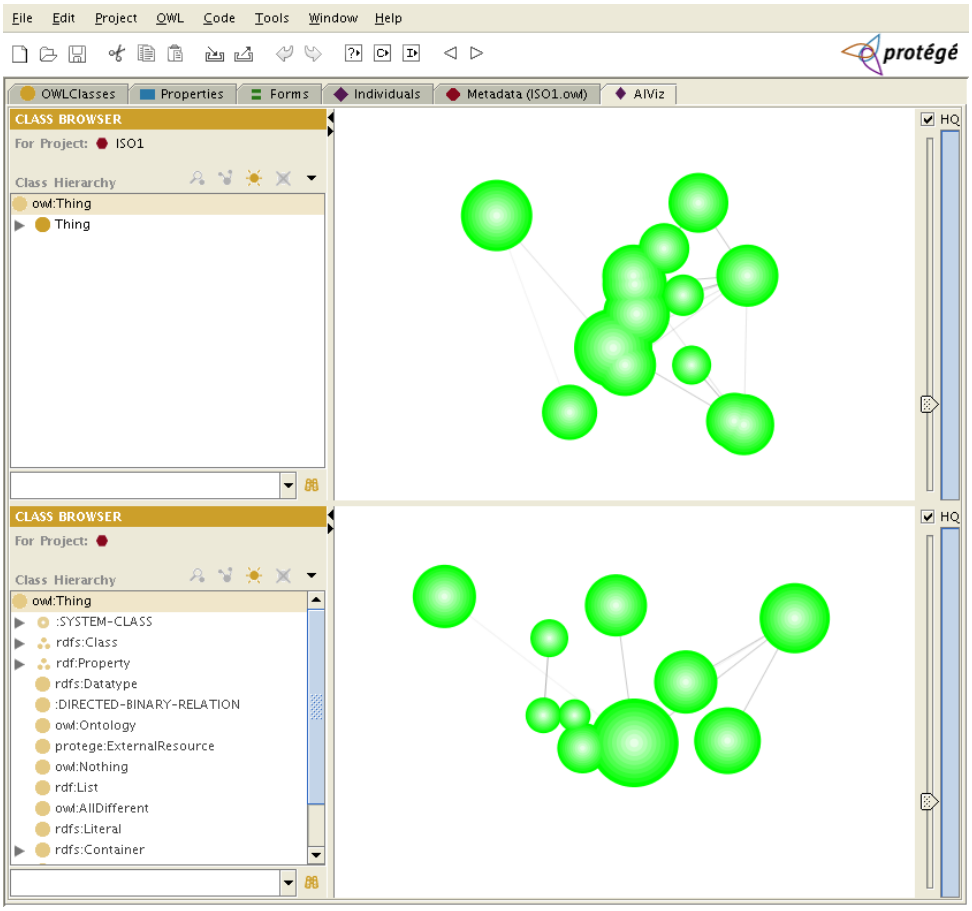

Figure 4.3: The startup screen of AlViz.

a black shading around the specified node. Simultaneously, the adjacent J-Tree gets expanded, unveiling the corresponding entity within the class browser, and highlighting the selected class. The name of the selected class and the configuration of its neighbourhood is disclosed in the class browser.

The final step of the implemented linking and brushing method unites both ontology panels. If the selected node is found to have a similar node in the opposed ontology the individual visual representations are linked together. In this use case (Figure 4.6) the selected node in the first ontology is labelled *Multidimensional_object*. The according node in the second ontology with the closest semantic meaning is *Multidimensional_number*, which has been selected and highlighted in both class browser and small world graph of the second ontology panel. Notice, that the class browser of the second ontology has been expanded to unveil the location of the appropriate class name.

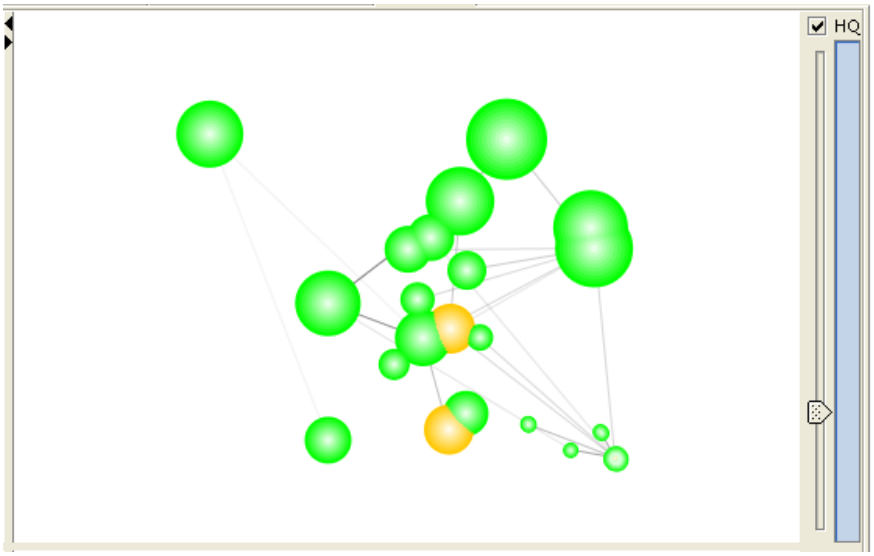

Figure 4.4: Temporary dissection of cluster nodes using the focal tool.

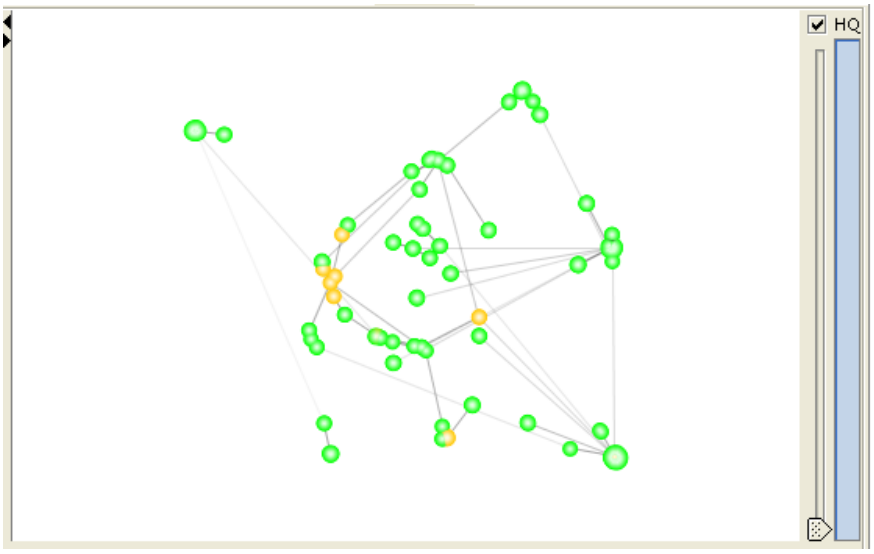

Figure 4.5: A fully expanded cluster graph.

In order to conduct the validation of the proposed ontology alignment mappings with confidence, the user has to be presented with complete information about the properties of the eligible nodes. For this cause, the icons in the upper part of the class browser frame are intended. Hovering above the icons reveals their meaning. The first icon (*View Class*) opens a new window that shows the Protégé class editor

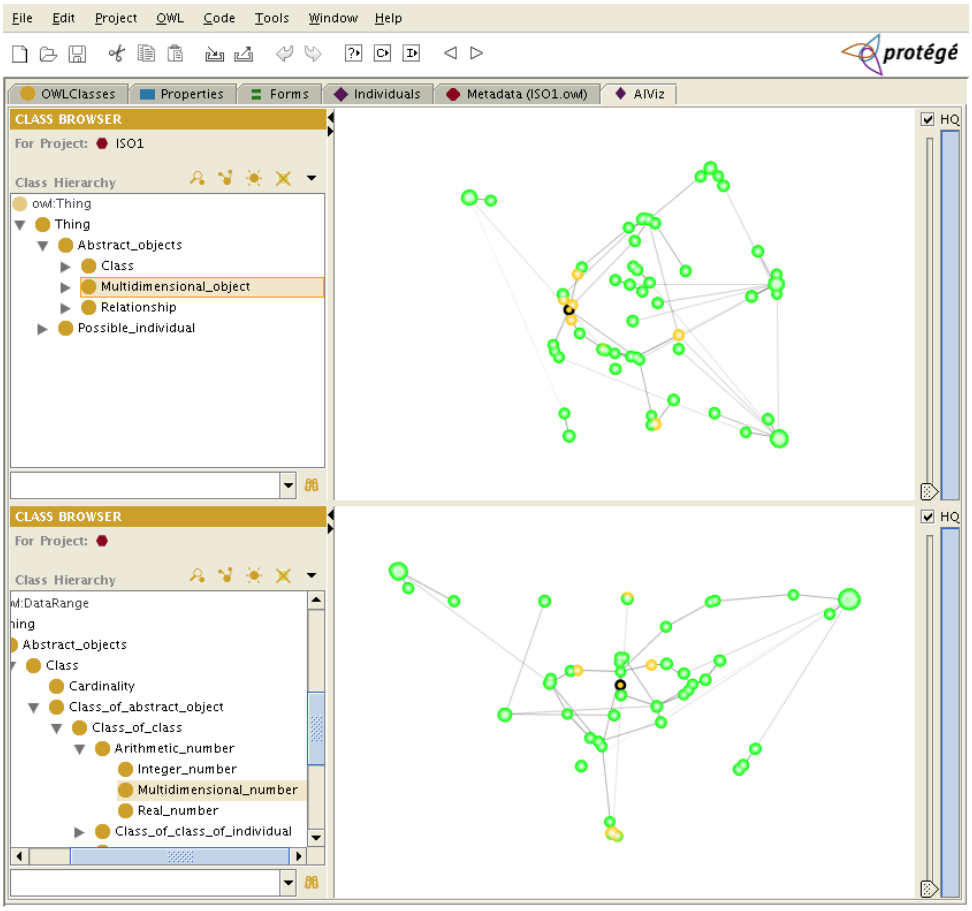

Figure 4.6: Illustration of linking and brushing in AlViz.

(Figure 4.7) of the indicated node. The class editor shows the specific configuration of a single ontology class. All properties, relations, conditions, restrictions, and annotations can be examined and modified in this view.

The second icon in the class browser heading (*View References To Class*) reveals the relations and role restrictions of the class node. It shows whether the examined entity has sub and super classes, the inherited properties, restrictions on the slot and facet properties, and whether direct instances are allowed.

The final two icons control the creation and deletion of single ontology classes. Activating the *Create Class* icon introduces a new ontology class as child node of the selected class. *Delete Class* does the opposite: Removing the selected class and the complete sub tree emerging from that node.

This concludes the survey on a use case using AlViz to validate ontology alignment results. Notice, that the work flow resembles vital parts from the task description

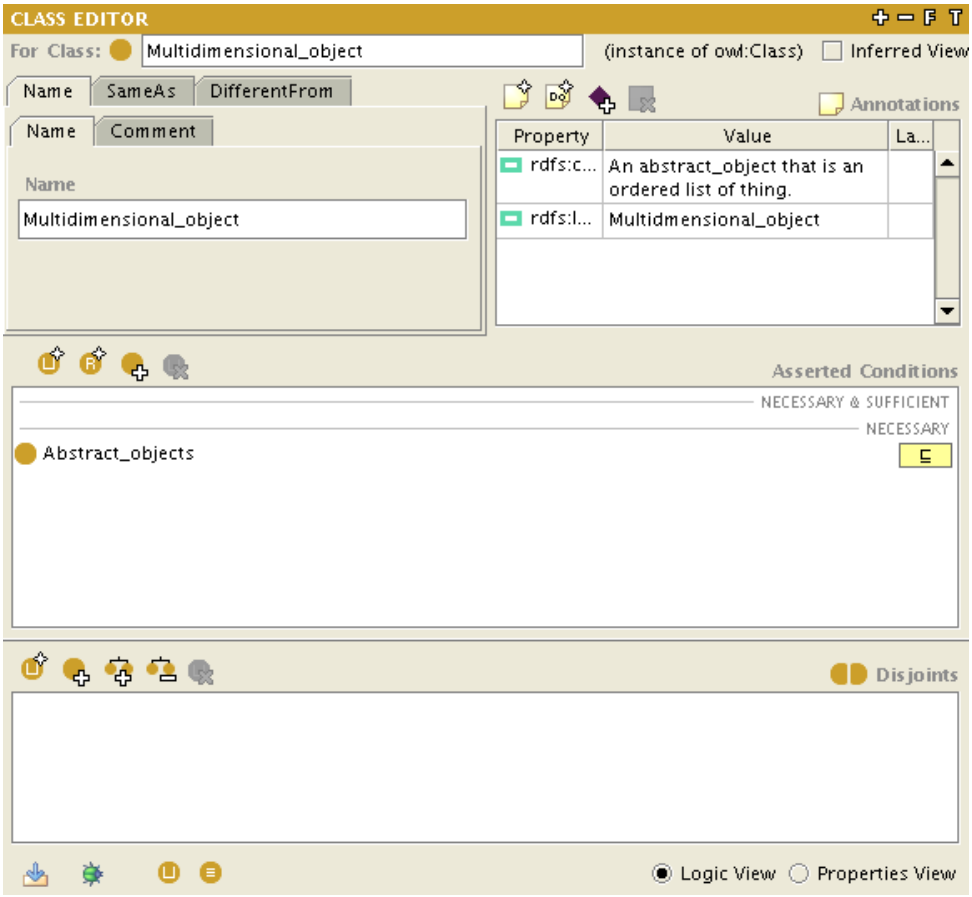

Figure 4.7: The Protégé class editor.

outlined in Section 3.1, thus qualifying as a proposal for the problem domain. In the following section we shall discuss the findings and evaluate the usability of AlViz as a visual ontology alignment tool.

4.2 Semantic Implications

The cluster method is laid out straight forward: When zooming out of the graph nearby nodes get merged and form a larger sphere. However, these emerging cluster nodes do not correspond to any specific entity in the ontology structure. Intuitively they seem to resemble a parent concept of the merged nodes, but if we recall the graph algorithm, every class of the ontology is rendered as single node in the visualisation. So are the parent nodes that are equally distributed and rendered as single nodes at the highest granularity.

This draws us to the conclusion, that the hierarchical structure of the ontology is not appropriately represented in the graph layout.

According to the small world algorithm, all nodes are placed on a plane, where the distance between the nodes is determined by the weight derived from the *is-a* relation of the ontology taxonomy. Therefore, classes are placed in the neighbourhood of their parents. As a consequence, during the cluster calculation, concepts and their sub-concepts are likely to be merged. The thus emerging node, however, does not represent either of them.

So, what are the semantic implications of a cluster node? In the case that a cluster node comprises a complete sub-tree — that is, a single concept and all subsumed subclasses of this concept — we can argue that the cluster node represents the generalised parent class. However, the clustering algorithm does not enforce the clustering of complete ontology sub-trees. We therefore have no knowledge about the semantic correspondence of a merged cluster node.

Actually, we believe this uncertainty to be a major asset for the task of analysing ontology alignments. We argued in Section 2.2.6 that the verification of alignment results can only be sophisticatedly achieved by human supervision. Only the human mind is capable of bridging the semantic gap that arouses when we try to map real world objects to formal data structures. Also, from Section 2.3 we know that an enormous quantity of information can be rapidly interpreted by the human visual system, and that it can discern patterns that are exceedingly difficult for machines to perceive — patterns that lie hidden in the visual representation of a graph layout.

We therefore argue, that the semantic implications of the clustered nodes can be identified by the user, even though we are still unable to identify their meaning. This visualisation method may promote insight by revealing inherent structural anomalies that remain hidden in an hierarchical correct representation. Of course, it would be feasible and easily implementable to enforce the clustering of siblings to emerge into generalised parent clusters. However, we like to stimulate the user to think *outside the square*.

4.3 Conceptual Evaluation

In Section 3.1 we defined the task description of visual ontology alignment and documented a number of requirements that have to be fulfilled in order to obtain satisfactory result. We will now revise the work flow of AlViz in order to validate its usefulness according the laid out requirements.

An independent evaluation can be conducted within a controlled user experiment. Since visualisation aims to enhance visual cognition in the human mind by developing insights from collected data for decision making support, the only valid evaluation technique is to involve a human. However, as cognition and insight can

not be measured decisively, we have to introduce performance as task indicator. The actual realisation of such a study exceeds the scope of this thesis, but we can outline the regulations and requirements of execution.

Our proposed evaluation design can be divided into three major validation categories: Validation of alignment proposal, usability, and technical performance. We shall describe each category in detail in the following sections.

4.3.1 Validation of Alignment Proposals

The hypotheses on the topic of validation and correction of alignment proposals have already been defined in the user study conducted by Sampson. The detailed description of the premisses and execution of the experiment can be found in [Sampson, 2007]. Sampson proposed two hypotheses to be evaluated:

H1 AlViz users perform better at the task of finding alignments missed by the automated tool, compared to those users without AlViz.

H2 AlViz users perform better at the task of accepting or rejecting questionable alignments, compared to those users without AlViz.

The test set up comprised two different ways to examine the alignments. First, by using AlViz to evaluate the validity and completeness of the ontology alignment results; and second, by using only Protégé with a list of concept definitions. In both cases, the participants were provided with a list of candidate alignments, compiled by an automated mapping tool. Various conditions have been counter-balanced to improve the internal validity of the obtained findings.

The results of this experiment showed a promising conclusion. “A number of the participants noticed that it was significantly easier to discover new alignments when using AlViz”, which confirmed hypothesis **H1**. Even though no significant results have been obtained to accept hypothesis **H2**, an analysis of each questionable alignment indicated, that AlViz users at least were able to correctly reject candidate alignments more often than Protégé users.

This first user study of a very early AlViz version shows that users appreciate the concept and design of the tool and benefit from the provided features to increase the performance of ontology alignment. We will therefore continue where Sampson left off and describe a follow up survey to evaluate the performance of AlViz.

One major problem that occurred during the first experiment was the initial phrasing of the second hypothesis. The quantitative analysis of the results revealed a semantic disparity of the expected and obtained findings. The question whether to accept or reject questionable alignments can only be adequately answered, if we a priori know the correct mapping. However, this decision is not easily made. We argue in

Section 2.2.6 that mismatches on the human level are likely to occur due to different interpretations of the real world. Accordingly, the alignment of ontologies falls for the same principles and introduces similar conclusions. The correctness of alignments are based on the subjective view of the users and can not be verified without bias by a formal and objective process. Thus, the alignments could only be validated in accordance with individual or cooperative agreement. We agree with the proponent that the use of golden standard ontologies does not reflect the intended purpose of ontology alignment tools, but we argue that they may be necessary to assure an independent and impartial experiment set up.

4.3.2 Usability

The target audience of AlViz is not necessarily the group of ontology experts. We would enjoy to lower the scientific requirements for users of ontology alignment tools and provide means to enable end users of knowledge engineering and semantic web developers the benefits of ontology integration. For this cause, the emphasis on usability is one of our major development concerns.

We have identified three requirements that deal with the user experience of visual ontology alignment tools and that should be tested in the evaluation process. The according test hypotheses can be phrased as follows:

H3 AlViz users perform better at comprehending the ontology structure to locate and identify previously specified alignments, compared to those users without AlViz.

H4 AlViz users perform better at distinguishing between different types of similarity, compared to those users without AlViz.

H5 AlViz users perform better at exploring and analysing the structural neighbourhood of specific concepts, than those users without AlViz.

The test set up for **H3** would imply the preparation of two ontologies and a list of aligned entities that are correctly marked as equal in both instances. The user is given the task of identifying the entities and describe their location in both ontologies. A successful completion of this task may indicate that the combination of visualisation techniques support untrained users to comprehend the ontology structure and the representation of semantic equality between the ontology proves to be self-explanatory

The participant is equipped with a list of alignment candidates and needs to express the type of similarity of the given concepts. This constitutes the premise of **H4**. The performance of the user inquiry indicates the ability of AlViz to render different types of similarity explicit.

In the configuration of **H5** the adequacy of AlViz to reveal the structural neighbourhood of an examined node is tested. The user is supplied with a list of concepts and has to give answer to questions about the neighbourhood of these classes. Such questions include, e.g. the name of super class and all sub classes; the similarity type of the super class of the aligned entity in the opposed ontology; whether the sub classes of the aligned entity in the second ontology are completely aligned, and if not, is there a missed alignment in the neighbourhood of the current entity that resembles a similarity? A good performance in this task confirms the capabilities of AlViz to efficiently visualise the neighbourhoods of aligned entities.

A satisfactory overall performance on these experiments indicates convincing design decisions in the development of AlViz to support the user with an adequate usability.

4.3.3 Technical Performance

From the requirements of visual ontology alignment tools we are able to identify two major technical performance indicators that need to be addressed in the design of AlViz. AlViz aims to provide support for the visualisation and alignment of large real world ontologies. Thus, the visual scalability is a major concern. The increase of information entropy arises from two facts: First, we are dealing with two ontologies. Compared with general ontology visualisation techniques, we already doubled the information load by introducing a second ontology. Second, the information about proposed alignments between these two ontologies needs to be visualised explicitly. Moreover, with every additional node the data complexity increases significantly, because the interrelations of the new nodes add vital information about the introduced relations to the ontology structures.

We argue, that the graphical representation with the layout of a clustered small world graph provides reasonable methods to allow an appropriate rendering of a large number of nodes. The interactive fish-eye lens technique that allows clusters to fold and unfold within the diameter of the focal tool, reveals only those characteristics of the graph that are currently observed, leaving the remaining layout untouched. We believe that this approach — in combination with the overall adjustable zoom level — avoids cluttered views.

However, this technical aspect of space constraints of visualisation patterns is primarily related to the visual representation of a single ontology. The comparison of ontology visualisation methods is not in the scope of this evaluation.

The evaluation of visual scalability of ontology alignment visualisation is yet an unexplored research field. The interpretation of the suitability of a specific layout representation is subject to the users observation and expectation in the interaction with the user interface. We therefore recommend a qualitative analysis on this sub-

ject in order to obtain a representative number of user statements to document their experiences with the visualisation technique.

The second technical performance indicator has been identified to be run-time complexity. The interactive approach requires intermediate response on arbitrary user input. The implementation of the validation process in AlViz reveals structural information in the interactive exploration process attended by the dynamic drawing of graph visualisations. A static display of the visual representations does not suffice to appropriately discern the ontology mapping information. It is therefore inevitable to keep the algorithm complexity on a reasonable level. The graph visualisation algorithm used in AlViz is based on the implementation by Ingram [2005], which actively reduces the computational complexity by taking invisible nodes out of the calculation. In a clustered rendering only $O(log(N))$ nodes are visible, thus efficiently reducing the computational overhead.

Obviously, this performance gain has to be put into perspective by the simultaneous display of four distinct visual representations in the overall view of AlViz. Of course, the two small world visualisations contribute significantly to a decrease in performance, whereas the rendering work load of the two J-Trees may be neglected.

It is difficult to design an evaluation experiment for this performance indicator. A technical comparison proves to be infeasible as there are no comparative visual ontology alignment tools. For the same reason, a qualitative study, is not likely to produce significant results, because we are not comparing tool performance but user experience of the interactive visualisation. We are therefore left with a qualitative user study to derive conclusion.

The configuration of the qualitative study would imply an extended use case scenario where the participants have to excessively use the available information visualisation methods to conclude the prepared questionnaire. However, the users have to be made aware that the start up time of the initial rendering of the visualisations has to be evaluated separately. In the aftermath of the experiment the participants need to be interviewed about their experience with the usability and responsiveness of the tool.

We argue, that the results of these experiments will greatly enhance the understanding of user interaction with AlViz and lead to appropriate development decisions to further increase the user experience.

4.4 Future Development

We are fully aware that the current prototype version of AlViz is less than complete. However, even in this early development stage the application proves to perform better at validating ontology alignment results than comparative tools [Sampson, 2007]. This indicates that the conceptual design and implementation are suitable to

exceed current approaches in ontology alignment by providing a visual representation of proposed alignments to enable understanding, agreement, and validation of the mappings.

We will indicate methods and changes to AlViz that should be implemented in future versions with an indented paragraph, marked with a vertical rule at the beginning.

First of all, the work flow is not complete. Following the task description in Section 3.1, the application has to be centred within the overall task. The tool needs to be the central hub of all activities and provide a congruent user interface to all sub tasks. In the current version the alignment results are assembled by an external mapping tool and subsequently supplied to be analysed by AlViz. This step happens before the application is initialised.

> In future versions the mapping process is triggered from within the application. The actual mapping results are still provided by external tools, but the user is presented with a configuration window to specify the necessary parameters. The initialisation and the workings of the mapping tool remain hidden.

This specific enhancement has been discussed and implemented by Gradwohl [2009].

The preparation of alignment results also happens only once at the beginning of the work flow. For the remainder of the validation process the user is left with a static view of aligned ontologies. This obstacle significantly reduces the interactive possibilities and hinders re-evaluation of confirmed findings. The user should be enabled to validate proposed alignments, confirm or reject appropriate mappings, and re-initialise the automated mapping process to double check the validation.

> In future versions of AlViz the user is able to initialise a re-calculation of automated alignment results on demand.

The embedding of automated mapping algorithms and the user initiated re-calculation of these mappings has been implemented by Gradwohl [2009].

In order to complete the validation process the user needs to explicate his/her findings. Currently the user has to keep track of validation results by external means; that is, manage the list of correctly aligned concepts separately and manually re-integrate these corrections into the mapping result file. This parallel task tends to be tedious and error prone, although it can easily be integrated to be supported by an automated process.

> In future versions AlViz supports the process of tracking validation results and re-integrates them into the automated alignment listing. The user either confirms or rejects a proposed alignment. Additionally, an alternative similarity relation can be assigned. AlViz manages the collection of confirmed, rejected, or corrected alignments internally. The graph visualisations are subsequently updated on the basis of the corrected listing to provide an up-to-date representation of ontology alignment results.

After the development of these enhancements the work flow can be completed by permanently storing the obtained alignment results. However, this particular modification needs some consideration. In order to retain independence of the application framework (i.e. Protégé in the current version) the decision to maintain an independent file to store the configuration of AlViz has been confirmed. At the time, the discussion about the contents and format of the file storage is not concluded yet. The debate at hand includes such topics as whether to keep one large file or multiple independent files; whether to store the initial alignment data and successively achieved results separately or to keep only the last accurate alignment mapping; how to encode various properties of the selected layout; and how to represent necessary interrelations to the environment provided by Protégé. Nevertheless, it is indisputable that the specification and release of an appropriate save file to store the progress of the validation progress has to be implemented.

> In future versions of AlViz the validation process is traced and permanently stored within a separate project file – or a defined set of various project files. The progress of a single validation session is preserved entirely to enable the user to restore an unfinished process and to continue with the operation at the point where s/he left.

Automated mapping algorithms usually work with thresholds to decide whether an examined pair of nodes are considered to be similar or not. These thresholds are defined either by the developers of the tool or can be adjusted by a parameter specified by the end user. In both cases they are globally defined and bear a significant chance to miss some vital similarities. For that matter, FOAM implemented an additional threshold to indicate results that indicate questionable alignments. That is, if the mapping algorithm computes a similarity value that lies between these two thresholds (i.e. below the first but above the second) these nodes are added to the collection of questionable alignments. In a nutshell: The algorithm is unable to autonomously decide whether the compared nodes are similar or not, and asks the user for confirmation. The FOAM algorithm stores these alignments in a separate file, which is ignored in the current version of AlViz.

> In future versions the user is provided with questionable alignment proposed by the automated mapping algorithm. The mappings are explicitly visualised and need to be confirmed or rejected by the user.

At the highest zoom level of the small world graph the node decomposition is not complete, still showing composed clusters. We believe that the transition of granularity level should be adjusted to reveal every single concept node at the highest zoom level. This shortcoming omits the display of a detailed view, which is necessary to explore the neighbourhood of an examined class.

> In future versions of AlViz the small world graph shows the complete decomposition of ontology structure. When pulling the zoom level slider to the lowest position (i.e. displaying the highest amplification) the cluster visualisation decomposes all merged nodes to reveal disjunct concepts.

The complete decomposition of the ontology nodes has been implemented by Huber [2009].

In the current stage of development AlViz proves to be useful for identifying and validating ontology alignment results. However, to provide a tool to actively support the task of visual ontology alignment some vital features and performance issues have to be addressed. The collection of enhancements and future improvements listed in this section are a first step in the completion of this task. Huber [2009] conducts a user evaluation where parts of the enlisted enhancements have already been implemented. AlViz is still a work in progress but the user response seems promising. The evaluation also reveals new insights about usability and possible improvements, which will be considered in future versions. We are confident to develop a soon to be released version that incorporates the proposed additions and deliver an application to fulfil the user's expectations.

4.5 Summary

In this chapter we presented a work flow scenario for visual ontology alignment conducted with our tool AlViz. It has been shown that the validation of ontology alignment proposals can be confirmed with the use of the current prototype version. However, to incorporate the task of visual ontology alignment — whereof validation of alignment results is a substantial subtask — further development is necessary.

We outlined a detailed evaluation strategy in order to evaluate the usefulness of AlViz according three major task requirements: Validation of alignment proposals, usability, and technical performance. The evaluation of the first requirement has already been analysed in a survey conducted by Sampson [2007]. The remaining two requirements are still open for evaluation.

Finally, we outlined the ongoing development process and identified crucial improvements to provide the user with a full fledged application to support visual ontology alignment.

Chapter 5

Conclusion

This thesis has been compiled to discuss the novel approach of ontology alignment visualisation. In this final chapter we will summarise our implementation of a visual ontology alignment validation and give an outlook of future plans, to develop a visual ontology alignment methodology.

5.1 Summary

The problem of ontology alignment is omnipresent in the current state of Semantic Web research. Many applications, automated agents, and human end users rely on the mapping of different ontologies to enable interoperability and knowledge exchange. Whereas automated alignment algorithms have been developed to provide promising results, a human ontology user will always be needed to bridge the semantic gap; that is, supervise the process and correct the proposed results.

Our contribution to alleviate the validation of ontology alignment results is the specification of a uniform ontology alignment format, to enable interoperability between ontology mapping algorithms and validation tools. We believe, that the use of a well-formed XML format to annotate mapping results will provide the Semantic Web community with appropriate means for archiving, distribution, comparison, and modularisation of ontology alignments. The current de facto standard of alignment result format has been provided by [Euzenat, 2004]. We argue, that our proposal exceeds the capabilities of the alignment API and facilitates a more granulated specification of alignment results. We aim to incorporate our extensions in the standard alignment format and propagate its usage to promote a common specification for ontology alignment results.

We use information visualisation techniques for the task of ontology alignment to present a novel approach of visual ontology alignment. The basic idea is founded

on the human visual capabilities to discern patterns in graphical representations that act as an external artefact to promote cognitive processing tasks. Under this premise we introduce a task description and outline requirements for visual ontology alignment. We argue that a uniform application framework is required to be the centre of the visual ontology alignment process. The application needs to be tailored as the central hub for all subsequent alignment steps, including the retrieval of preliminary mapping results, the visualisation of the compared ontologies, the presentation of candidate mappings, the validation and correction of proposed alignments, and the initialisation and integration of re-calculated mapping results. To complete the work flow the application has to facilitate the permanent storage and distribution of the final alignment process.

To put the methodology on trial we present our tool, AlViz, which has been developed with the aim to facilitate the task of ontology alignment by providing a framework for visualisation and verification of ontology alignment results. AlViz is designed as multi-view plug-in for Protégé using J-Trees and small world graphs. A total number of four visualisations are provided to allow the user to freely explore the ontologies in detail. Much effort has been taken to incorporate a cluster visualisation of the ontology hierarchy and annotate the graph with alignment results. Based on the similarity relation of compared nodes, the appropriate cluster nodes are colourised. One of the main features of the visualisation proves to be the linking and brushing technique. Selected nodes reveal their similarity relation by highlighting the appropriate nodes in all four ontology representations.

The ultimate goal is to develop a tool to facilitate visual ontology alignment. In the current version the work flow has not been completed yet. However, important parts of the task requirements are implemented such that an important subtask can be performed: the validation of ontology alignment results. We obtained encouraging results with this early prototype and continue to implement the outlined development strategy.

We discuss a formal evaluation of the capabilities and the usability of AlViz. In conducting the outlined experiment strategies we will be able to verify the implemented methodology. Finally, we identify the shortcomings of the current prototype version and discuss future improvements. Obviously, an important milestone is the work flow extension to incorporate the task of visual ontology alignment. The detailed enhancements include the initialisation of external mapping algorithms, interactive correction of proposed alignments, and the storage of final alignment results. The development progress is accompanied by continuous evaluation and user studies [Huber, 2009; Gradwohl, 2009].

We are confident to provide a comprehensive framework for visualisation and verification of ontology alignment results that can reasonably be extended into a full fledged visual ontology alignment framework.

5.2 Closure

The need for a visual ontology alignment tool has been driven by the lack of human readable and interpretable ontology mapping results. Whereby significant progress has been made in the development of automated mapping algorithms, the interpretation of the mapping results remain difficult to read for a human.

The decision to visualise the ontologies and overlay the mapping results in an appealing and informative way comes naturally. Unfortunately, this specific scenario has not been approached and discussed yet in ontology literature. Accordingly, no surveys about appropriate visualisation methodologies for the specific task of ontology alignment visualisation could have been consulted. Therefore the question whether a specific visualisation method is suitable to capture and present the encoded information about relations between two ontologies had to be investigated before we could commence the development of a visualisation tool.

The use of a clustering algorithm seemed feasible for a hierarchical data structure. However, during this discussion it became obvious, that ontologies embody more than the taxonomic *is-a* relation. Moreover, ontologies tend to have a large number of relations that bear important semantic information about the domain of interest. From a graph theoretic point of view, this property suddenly introduces a vast number of edges between a series of nodes, causing a significant decrease of the average path length of the graph. This characteristics led us to focus on small world graphs. Even though we decided to restrict the visualisation to capture only the inheritance relation in a first prototype version, the chosen graph layout is powerful enough to adapt to a large number of relations. In future versions of AlViz we will be able to exploit the small world graph visualisation to render multiple relations.

To speed up the development process we decided to exploit and extend existing solutions. The use of Protégé as a platform has been triggered by its unique status among the Semantic Web and knowledge engineering communities. The open design and mature code base offered a perfect framework for the rapid development of a prototype. Nevertheless, the combination of automated mapping algorithms and visualisation methods introduced unforeseen problems during the integration process. The unstructured representation of mapping results forced us to develop a structured and well-defined specification. This decision proved to be essential in the subsequent development process. The XML based format allowed us to quickly locate and identify specific nodes and relations in the mapping file. In this way we were able to verify the correct rendering of the visualisation and pinpoint bugs efficiently.

The incorporation of the small world visualisation algorithm has to been seen as the major asset of AlViz. I would like to thank Stephen Ingram for his excellent work in the implementation of the algorithm, and for letting us use his code base. The integration of the graph algorithm proved to be the easiest part of the visualisation

development. As so often, the tricky parts are not immediately evident in the end result but lay hidden in the background mechanism. The specification of a suitable data structure, the colourisation of individual nodes and clusters, the decomposition of the mapping results — just to name a few stumbling blocks that delayed the development.

Albeit the obstacles we compiled a first prototype of AlViz that marks an essential milestone for future development. The basic structure has been defined and implemented to facilitate future enhancements to focus on specific topics. Modifications of the underlying framework should not be necessary.

5.3 Future Work

The main concern of future development is, of course, the implementation of the full work flow environment. Our goal is to broaden the capabilities of AlViz from a validation tool for proposed mapping results to a full fledged visual ontology alignment tool. Thus, it is important to provide the user with an unified user interface to embed all necessary subtasks. We outlined the according development issues in Section 4.4.

After this crucial transformation step, many extensions are possible. For example, the integration of different mapping tools. The user is provided with a selection menu to choose a desired mapping algorithm. This may promote the alignment quality by convenient adjustment of algorithm parameters and promote comparison of competitive mapping results.

For the representation of ontology alignment results we developed a comprehensive XML specification to facilitate interoperability between ontology mapping and ontology authoring tools. In our research we identified a conceptual similar proposal for the layout of a alignment results format. However, the specification of Euzenat [2004] did not meet the necessary requirements for the integration into AlViz. In a consolidation process Gradwohl [2009] extends the format design of Euzenat to appropriately reflect the degree of detail to express all alignment features contemplated by AlViz. We hope to contribute to a wider acceptance of the alignment format to promote compatibility and interaction between ontology authoring tools.

Appendix

Listing A-1: AlViz XSD

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <xsd:element name="AlViz">
    <xsd:complexType mixed="false">
      <xsd:all>
        <xsd:element name="Nodes" type="NodesType" minOccurs="1" maxOccurs="1"/>
      </xsd:all>
    </xsd:complexType>
  </xsd:element>

  <xsd:complexType name="NodesType">
    <xsd:choice minOccurs="1" maxOccurs="unbounded">
      <xsd:element name="Node" type="NodeType" />
    </xsd:choice>
  </xsd:complexType>

  <xsd:complexType name="NodeType">
    <xsd:choice>
      <xsd:element name="Entity" type="EntityType" minOccurs="1" maxOccurs="1" />
      <xsd:element name="CNode" type="CNodeType" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:choice>
    <xsd:attribute name="uri" type="xsd:anyURI" />
  </xsd:complexType>

  <xsd:complexType name="CNodeType">
    <xsd:choice>
      <xsd:element name="Entity" type="EntityType" minOccurs="1" maxOccurs="1" />
      <xsd:element name="Similarity" type="SimilarityType" minOccurs="0" maxOccurs="
          unbounded" />
      <xsd:element name="SimilarityInfo" type="SimilarityInfoType" minOccurs="1"
          maxOccurs="1" />
    </xsd:choice>
  </xsd:complexType>

  <xsd:complexType name="EntityType">
    <xsd:attribute name="label" type="xsd:string" />
    <!-- type is one of
      C   concept
      P   property
      I   instance
      D   data property
      O   object property
      -->
    <xsd:attribute name="type" type="xsd:string" />
  </xsd:complexType>

  <xsd:complexType name="SimilarityInfoType">
    <xsd:all>
      <xsd:element name="Confidence">
        <xsd:attribute name="value" type="float" />
      </xsd:element>
      <xsd:element name="Correct">
```

```
        <!-- true/false or 1/0 -->
        <xsd:attribute name="value" type="boolean" />
      </xsd:element>
    </xsd:all>
  </xsd:complexType>

  <xsd:complexType name="SimilarityType">
    <!--  rule is one of
     -1   not similar
      0   Syntactic similarity
      1   Equal URI
      2   Similar Superclass
      3   Similar Subclass
      4   Similar Class Data Properties
      5   Similar Class Object Properties From
      6   Similar Class Object Properties To
      7   Equal Class Member Individuals
      8   Narrower Than
      9   Broader Than
     10   Similar Data Property Domain
     11   Similar Super Data Properties
     12   Similar Sub Data Properties
     13   Similar Data Property Members
     14   Similar Object Property Domain
     15   Similar Object Property Range
     16   Similar Object Property Super
     17   Similar Object Property Sub
     18   Similar Object Property Members
     19   Similar Individual Member of Class
     20   Similar Individual Data Properties
     21   Similar Individual Object Property Members From
     22   Similar Individual Object Property Members To
    -->
    <xsd:attribute name="rule" type="integer" />
    <xsd:attribute name="value" type="float" />
  </xsd:complexType>

</xsd:schema>
```

Listing A-2: AlViz DTD

```
<!DOCTYPE AlViz [
        <!ELEMENT AlViz (Nodes)>
        <!ELEMENT Nodes (Node+)>
        <!ELEMENT Node (Entity,CNode+)>
        <!ATTLIST Node uri CDATA>
        <!ATTLIST Entity label CDATA>
        <!ATTLIST CNode uri CDATA>
        <!ELEMENT CNode (Entity, Similarity*)>
        <!ATTLIST Similarity
                rule (-1|0|1|2|3|4|5|6|7|8|9|10|11|12|13|14|15|16|17|18|19|20|21|22)
                value CDATA>
        <!ELEMENT Similarity (confidence?, correct?)>
        <!ATTLIST confidence value CDATA>
        <!ATTLIST correct value CDATA>
]>
```

Table A-1: List of semantic relations

Category	Colour	Description
Equal	red	Equal URI
		Equal Class Member Individuals
Syntactically Equal Entities	orange	Syntactic similarity
Similar	green	Similar Superclasses
		Similar Subclasses
		Similar Class Data Properties
		Similar Class Object Properties From
		Similar Class Object Properties To
		Similar Data Property Domain
		Similar Super Data Properties
		Similar Sub Data Properties
		Similar Data Property Members
		Similar Object Property Domain
		Similar Object Property Range
		Similar Object Property Super
		Similar Object Property Sub
		Similar Object Property Members
		Similar Individual Member of Class
		Similar Individual Data Properties
		Similar Individual Object Property Members From
		Similar Individual Object Property Members To
Narrower than	magenta	Narrower Than
Broader than	blue	Broader Than
Not Found	yellow	Different

Acknowledgements

I am very grateful to Monika Lanzenberger for her supervision. She promoted the freedom I appreciated, and enforced the deadlines I needed.

My family has always been a great resource of support, chaos, and dedication. Their love and faith always kept me going. Special thanks to my little sister, who provided me with a deadline. Happy birthday, sweety.

I would like to thank Markus Rester and Stoiko Ivanov for all the fruitful discussions, brilliant ideas, legitimate criticism, and wonderful diners. They also helped with the thesis. Stoiko implemented the linking and brushing methods and enhanced the visualisation layout. Markus provided me with essential insights in the use of LaTeX.

Special thanks go out to Gerald Steinhardt, who encouraged me to look beyond scientific borders and to embrace my future aside the university.

Bibliography

[Adamic 1999] Adamic, Lada A.: The Small World Web. In: *ECDL '99: Proceedings of the Third European Conference on Research and Advanced Technology for Digital Libraries*. London, UK : Springer-Verlag, 1999, p. 443–452. – URL http://www.springerlink.com/content/4fjgx8c7m92nqe05/. – ISBN 3-540-66558-7

[Alpert and Hajaj 2008] Alpert, Jesse ; Hajaj, Nissan: *We knew the web was big...* July 2008. – URL http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html

[An et al. 2007] An, Yoo J. ; Geller, James ; Wu, Yi-Ta ; Chun, Soon A.: Semantic Deep Web: Automatic Attribute Extraction from the Deep Web Data Sources. In: *SAC '07: Proceedings of the 2007 ACM symposium on Applied computing*. New York, NY, USA : ACM, 2007, p. 1667–1672. – URL http://doi.dx.org/10.1145/1244002.1244355. – ISBN 1-59593-480-4

[Bergman 2001] Bergman, Michael K.: The Deep Web: Surfacing Hidden Value. In: *Journal of Electronic Publishing* 7 (2001), August, Nr. 1. – URL http://dx.doi.org/10.3998/3336451.0007.104

[Berners-Lee et al. 2001] Berners-Lee, Tim ; Hendler, James ; Lassila, Ora: The Semantic Web. In: *Scientific American* 284 (2001), Nr. 5, p. 34–43. – URL http://www.sciam.com/article.cfm?id=the-semantic-web

[Brin and Page 1998] Brin, Sergey ; Page, Lawrence: The Anatomy of a Large-Scale Hypertextual Web Search Engine. In: *Comput. Netw. ISDN Syst.* 30 (1998), Nr. 1-7, p. 107–117. – URL http://dx.doi.org/10.1016/S0169-7552(98)00110-X. – ISSN 0169-7552

[Card et al. 1999] Card, Stuart K. ; Mackinlay, Jock D. ; Shneiderman, Ben: Using Vision to Think. In: *Readings in Information Visualization: Using Vision to Think* 1 (1999), p. 579–581. – URL http://www.cs.umd.edu/hcil/pubs/books/readings-info-vis.shtml. ISBN 1-55860-533-9

[Chen 1976] Chen, Peter Pin-Shan: The entity-relationship model—toward a unified view of data. In: *ACM Trans. Database Syst.* 1 (1976), Nr. 1, p. 9–36. – URL http://doi.acm.org/10.1145/320434.320440

[Christopoulou et al. 2005] Christopoulou, Eleni ; Goumopoulos, Christos ; Kameas, Achilles: An Ontology-Based Context Management and Reasoning Process for UbiComp Applications. In: *sOc-EUSAI '05: Proceedings of the 2005 Joint Conference on Smart Objects and Ambient Intelligence*. New York, NY, USA : ACM, 2005, p. 265–270. – URL http://dx.doi.org/10.1145/1107548.1107613. – ISBN 1-59593-304-2

[Ehrig 2005] Ehrig, Marc: *Ontology Alignment: Bridging the Semantic Gap*, Fakultät für Wirtschaftswissenschaften der Universität Fridericiana zu Karlsruhe (TH), Ph.D. thesis, 12 2005. – URL http://www.aifb.uni-karlsruhe.de/WBS/meh/diss_meh.pdf

[Ehrig and Sure 2004] Ehrig, Marc ; Sure, York: Ontology Mapping - An Integrated Approach. In: *The Semantic Web: Research and Applications* Volume 3053/2004, Springer Berlin / Heidelberg, 2004, p. 76–91. – URL http://www.springerlink.com/index/T3M6YJDEB99FT7AK.pdf

[Ehrig and Sure 2005] Ehrig, Marc ; Sure, York: Foam - Framework for Ontology Alignment and Mapping; Results of the Ontology Alignment Initiative. In: *Proceedings of the Workshop on Integrating Ontologies* Volume 156, URL http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-156/paper11.pdf, 2005, p. 72–76

[Ernst and Storey 2003] Ernst, Neil A. ; Storey, Margaret A.: A Preliminary Analysis of Visualization Requirements in Knowledge Engineering Tools / University of Victoria. Victoria : University of Victoria, 2003. – Research Report. – URL http://neilernst.net/docs/pubs/ernst-survey.pdf

[Euzenat 2004] Euzenat, Jérôme: An API for Ontology Alignment. In: *The Semantic Web – ISWC 2004*, Springer, 2004, p. 698–712. – URL http://www.springerlink.com/content/dy8y9f31a9gt9762

[Euzenat et al. 2004] Euzenat, Jérôme ; Bach, Thanh L. ; Barrasa, Jesús ; Bouquet, Paolo ; Bo, Jan D. ; Dieng, Rose ; Ehrig, Marc ; Hauswirth, Manfred ; Jarrar, Mustafa ; Lara, Ruben ; Maynard, Diana ; Napoli, Amedeo ; Stamou, Giorgos ; Stuckenschmidt, Heiner ; Shvaiko, Pavel ; Tessaris, Sergio ; Acker, Sven V. ; Zaihrayeu, Ilya: State of the Art on Ontology Alignment / Institut National de Recherche en Informatique et en Automatique (INRIA). URL http://knowledgeweb.semanticweb.org/semanticportal/deliverables/D2.2.3.pdf, AUG 2004 (2.2.3). – Knowledge Web Deliverable

[Euzenat and Shvaiko 2007] Euzenat, Jérôme ; Shvaiko, Pavel: *Ontology Matching*. Springer, 2007. – 333 p. – URL http://book.ontologymatching.org/. – ISBN 978-3-540-49611-3

[Fekete et al. 2008] Fekete, Jean-Daniel ; Wijk, Jarke van ; Stasko, John ; North, Chris: The Value of Information Visualization. In: *Information Visualization* Volume 4950/2008. Springer Berlin / Heidelberg, 2008, p. 1–18. – URL http://www.springerlink.com/content/q255124278700854

[Gangemi et al. 1998] Gangemi, Aldo ; Pisanelli, Domenico M. ; Steve, Geri: Ontology Integration: Experiences with Medical Terminologies. In: Guarino, Nicola (Editor): *Formal Ontology in Information Systems*, IOS Press, 1998, p. 163–178. – URL http://www.loa-cnr.it/Papers/fois98r.pdf

[Gennari et al. 2003] Gennari, John H. ; Musen, Mark A. ; Fergerson, Ray W. ; Grosso, William E. ; Crubézy, Monica ; Eriksson, Henrik ; Noy, Natalya F. ; Tu, Samson W.: The evolution of Protégé: an environment for knowledge-based systems development. In: *International Journal of Human-Computer Studies* 58 (2003), Nr. 1, p. 89–123. – URL http://dx.doi.org/10.1016/S1071-5819(02)00127-1

[Gil and García 2004] Gil, Rosa ; García, Roberto: Measuring the semantic web. In: *AIS SIGSEMIS Bulletin* 1 (2004), p. 6–10. – URL http://rhizomik.net/~roberto/papers/rgrgmtsr2005.pdf

[Gonçalves et al. 2009] Gonçalves, Bernardo ; Zamborlini, Veruska ; Guizzardi, Giancarlo ; Filho, José G. P.: An ontology-based application in heart electrophysiology: representation, reasoning and visualization on the web. In: *SAC '09: Proceedings of the 2009 ACM symposium on Applied Computing*. New York, NY, USA : ACM, 2009, p. 816–820. – URL http://doi.acm.org/10.1145/1529282.1529456

[Gradwohl 2009] Gradwohl, Manuel: *Integration von Ontology Alignment Algorithmen in AlViz*, Technische Universität Wien, Master thesis, September 2009

[Grau 2004] Grau, Bernardo C.: A possible simplification of the semantic web architecture. In: *WWW '04: Proceedings of the 13th international conference on World Wide Web*. New York, NY, USA : ACM, 2004, p. 704–713. – URL http://doi.acm.org/10.1145/988672.988769

[Gruber 1993] Gruber, Thomas R.: Toward principles for the design of ontologies used for knowledge sharing. In: *International Journal of Human-Computer Studies*, Kluwer Academic Publishers, 1993, p. 907–928. – URL http://tomgruber.org/writing/onto-design.pdf

[Guarino 1997] Guarino, Nicola: Understanding, building and using ontologies. In: *International Journal of Human Computer Studies* 46 (1997), Nr. 2-3, p. 293–310. – URL http://dx.doi.org/10.1006/ijhc.1996.0091

[Gómez-Pérez 1994] Gómez-Pérez, Asunción: From Knowledge Based Systems to Knowledge Sharing Technology: Evaluation and Assessment / Knowledge Systems, AI Laboratory. URL ftp://ftp.ksl.stanford.edu/pub/KSL_Reports/KSL-94-73.ps.gz, 1994 (KSL-94-73). – Research Report

[Gómez-Pérez 2001] Gómez-Pérez, Asunción: Evaluation of ontologies. In: *International Journal of Intelligent Systems* 16 (2001), Nr. 3, p. 391–409. – URL http://www3.interscience.wiley.com/journal/77002631/abstract

[Gómez-Pérez and Benjamins 1999] Gómez-Pérez, Asunción ; Benjamins, V. R.: Overview of Knowledge Sharing and Reuse Components: Ontologies and Problem-Solving Methods. In: *Proceedings of the IJCAI-99 Workshop on Ontologies and Problem-Solving Methods (KRR5)*, URL http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-18/1-gomez.pdf, August 1999

[van Ham and van Wijk 2002] Ham, F. van ; Wijk, J.J. van: Beamtrees: compact visualization of large hierarchies. In: *IEEE Symposium on Information Visualization*, URL http://dx.doi.org/10.1109/INFVIS.2002.1173153, October 2002, p. 93–100

[van Ham and van Wijk 2004] Ham, Frank van ; Wijk, Jarke J. van: Interactive Visualization of Small World Graphs. In: *IEEE Symposium on Information Visualization* 0 (2004), October, p. 199–206. – URL http://doi.ieeecomputersociety.org/10.1109/INFVIS.2004.43

[Hendler 2001] Hendler, J.: Agents and the Semantic Web. In: *Intelligent Systems, IEEE* 16 (2001), Mar-Apr, Nr. 2, p. 30–37. – URL http://dx.doi.org/10.1109/5254.920597

[Horrocks 2007] Horrocks, Ian: Semantic web: the story so far. In: *W4A '07: Proceedings of the 2007 international cross-disciplinary conference on Web accessibility (W4A)*. New York, NY, USA : ACM Press, 2007, p. 120–125. – URL http://doi.acm.org/10.1145/1243441.1243469

[Huber 2009] Huber, Martin: *Visuelle Unterstützung des Ontologie-Alignments*, Technische Universität Wien, Master thesis, May 2009

[Ingram 2005] Ingram, Stephen F.: An Interactive Small World Graph Visualization / University of British Columbia. URL http://www.cs.ubc.ca/~sfingram/cs533C/small_world.pdf, 2005. – Research Report

[Katifori et al. 2007] Katifori, Akrivi ; Halatsis, Constantin ; Lepouras, George ; Vassilakis, Costas ; Giannopoulou, Eugenia: Ontology visualization methods—a survey. In: *ACM Computing Surveys* 39 (2007), Nr. 4, p. 10. – URL http://doi.acm.org/10.1145/1287620.1287621

[Kauppinen and Hyvönen 2004] Kauppinen, Tomi ; Hyvönen, Eero: Bridging the Semantic Gap between Ontology Versions. In: *Proceedings of the 11th Finnish AI Conference, Web Intelligence Symposium* Volume 2, Finnish Artificial Intelligence Society, September 2004, p. 63–72. – URL http://www.seco.tkk.fi/events/2004/2004-09-02-web-intelligence/papers/changebridges-wis04.pdf

[Klein 2001] Klein, Michel: Combining and Relating Ontologies: An Analysis of Problems and Solutions. In: Gomez-Perez, A. (Editor) ; Gruninger, M. (Editor) ; Stuckenschmidt, H. (Editor) ; Uschold, M. (Editor): *Workshop on Ontologies and Information Sharing, IJCAI'01*. Seattle, USA, 2001. – URL http://www.cs.vu.nl/~mcaklein/papers/IJCAI01-ws.pdf

[Krovetz and Croft 1992] Krovetz, Robert ; Croft, W. B.: Lexical Ambiguity and Information Retrieval. In: *ACM Transactions on Information Systems* 10 (1992), April, Nr. 2, p. 115–141. – URL http://doi.acm.org/10.1145/146802.146810. – ISSN 1046-8188

[Lanzenberger and Sampson 2006] Lanzenberger, Monika ; Sampson, Jennifer: AlViz - A Tool for Visual Ontology Alignment. In: *IV '06: Proceedings of the conference on Information Visualization*. Washington, DC, USA : IEEE Computer Society, 2006, p. 430–440. – URL http://doi.ieeecomputersociety.org/10.1109/IV.2006.18

[Lanzenberger and Sampson 2007] Lanzenberger, Monika ; Sampson, Jennifer: Human-Mediated Visual Ontology Alignment. In: *Human Interface and the Management of Information. Interacting in Information Environments* Volume 4558/2007, Springer Berlin / Heidelberg, August 2007, p. 394–403. – URL http://www.springerlink.com/content/e7980u513420l360/

[Lanzenberger et al. 2008] Lanzenberger, Monika ; Sampson, Jennifer J. ; Rester, Markus ; Naudet, Yannick ; Latour, Thibaud: Visual ontology alignment for knowledge sharing and reuse. In: *Journal of Knowledge Management* 12 (2008), Nr. 6, p. 102–120. – URL http://dx.doi.org/10.1108/13673270810913658. – ISSN 1367-3270

[Li et al. 2005] Li, Li ; Wu, Baolin ; Yang, Yun: Agent-based ontology integration for ontology-based applications. In: *AOW '05: Proceedings of the 2005 Australasian Ontology Workshop* Volume 58. Darlinghurst, Australia, Australia : Australian Computer Society, Inc., 2005, p. 53–59. – URL http://portal.acm.org/citation.cfm?id=1151943

[Madhavan et al. 2008] Madhavan, Jayant ; Ko, David ; Kot, Łucja ; Ganapathy, Vignesh ; Rasmussen, Alex ; Halevy, Alon: Google's Deep Web crawl. In: *Proceedings of the VLDB Endowment* 1 (2008), Nr. 2, p. 1241–1252. – URL http://www.vldb.org/pvldb/1/1454163.pdf

[McGuinness 2002] McGuinness, Deborah L.: Ontologies Come of Age. In: Fensel, Dieter (Editor) ; Hendler, Jim (Editor) ; Lieberman, Henry (Editor) ; Wahlster, Wolfgang (Editor): *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*. MIT Press, 2002. – URL http://www.ksl.stanford.edu/people/dlm/papers/ontologies-come-of-age-mit-press-(with-citation).htm

[Milgram 1967] Milgram, Stanley: The Small World Problem. In: *Psychology Today* 2 (1967), p. 60–67

[Mommers 2003] Mommers, L.: Application of a Knowledge -Based Ontology of the Legal Domain in Collaborative Workspaces. In: *ICAIL '03: Proceedings of the 9th International Conference on Artificial Intelligence and Law*. New York, NY, USA : ACM, 2003, p. 70–76. – URL http://doi.acm.org/10.1145/1047788.1047799

[Newman 2004] Newman, M. E. J.: Fast Algorithm for Detecting Community Structure in Networks. In: *Physical Review E* 69 (2004), Jun, Nr. 6, p. 066133+. – URL http://dx.doi.org/10.1103/PhysRevE.69.066133

[Ng 2000] Ng, Gary Kwok-Chu: *Interactive Visualisation Techniques for Ontology Development*, University of Manchester, Ph.D. thesis, November 2000. – URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.23.335

[Noack 2003] Noack, Andreas: An Energy Model for Visual Graph Clustering. In: *Graph Drawing* Volume 2912/2004, URL http://www.springerlink.com/content/bnkwtl1vanl222ln/, March 2003, p. 425–436

[Noack 2005] Noack, Andreas: Energy-Based Clustering of Graphs with Nonuniform Degrees. In: *Graph Drawing* Volume 3843/2006, URL http://www.springerlink.com/content/782176470300 2mw1/, January 2005, p. 309–320

[Noy and McGuinness 2001] Noy, Natalya F. ; McGuinness, Deborag L.: *Ontology Development 101: A Guide to Creating Your First Ontology*. 2001. – URL http://www.stanford.edu/~natalya/papers/ontology_tutorial.pdf

[Obrst et al. 2003] Obrst, Leo ; Liu, Howard H. ; Wray, Robert E.: Ontologies for corporate web applications.

In: *AI Magazine* 24 (2003), September, Nr. 3, p. 49–62. – URL http://www.aaai.org/ojs/index.php/aimagazine/article/viewArticle/1718

[Øhrstrøm et al. 2005] Øhrstrøm, Peter ; Andersen, Jan ; Schärfe, Henrik: What Has Happened to Ontology. In: *Conceptual Structures: Common Semantics for Sharing Knowledge* Volume 3596/2005. Springer Berlin / Heidelberg, 2005, p. 425–438. – URL http://www.springerlink.com/content/04kf69c3npayc8m8/

[Pinto and Martins 2001] Pinto, Helena S. ; Martins, ao P: A Methodology for Ontology Integration. In: *K-CAP '01: Proceedings of the 1st International Conference on Knowledge Capture*, ACM New York, NY, USA, 2001, p. 131–138. – URL http://doi.acm.org/10.1145/500737.500759

[Pinto and Martins 2004] Pinto, Helena S. ; Martins, João P.: Ontologies: How can They be Built? In: *Knowledge and Information Systems* 6 (2004), July, Nr. 4, p. 441–464. – URL http://www.springerlink.com/content/0p5yqrdh5t5dvd06/

[Sampson 2007] Sampson, Jennifer: *A Comprehensive Framework for Ontology Alignment Quality*, Norwegian University of Science and Technology, Faculty of Information Technology, Mathematics and Electrical Engineering Department of Computer and Information Science, Ph.D. thesis, 2007. – URL http://www.idi.ntnu.no/grupper/su/publ/phd/phd-sampson-mar07.pdf

[Sampson and Lanzenberger 2006] Sampson, Jennifer ; Lanzenberger, Monika: Visual Ontology Alignment for Semantic Web Applications. In: *Advances in Conceptual Modeling - Theory and Practice* Volume 4231/2006, Springer Berlin / Heidelberg, November 2006, p. 405–414. – URL http://www.springerlink.com/content/x314688512827551/

[Sampson et al. 2008] Sampson, Jennifer ; Lanzenberger, Monika ; Veres, Csaba: Facilitating Interoperability in Semantic Web Applications Using Ontologies. In: *Complex, Intelligent and Software Intensive Systems, International Conference* 0 (2008), p. 233–239. – URL http://doi.ieeecomputersociety.org/10.1109/CISIS.2008.107

[Sampson et al. 2007] Sampson, Jennifer ; Veres, Csaba ; Lanzenberger, Monika: An Integrated Approach to Organizational Data Interoperability. In: *Enterprise Interoperability, New Challenges and Approaches*. London : Springer London, 2007, p. 387–396. – URL http://www.springerlink.com/content/m884887710uj7268/. – Vortrag: Interoperability for Enterprise Software and Applications Conference, I-ESA'06, Bordeaux, France; 2006-03-22 – 2006-03-24

[Seki and Mostafa 2005] Seki, Kazuhiro ; Mostafa, Javed: An application of text categorization methods to gene ontology annotation. In: *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM New York, NY, USA, 2005, p. 138–145. – URL http://doi.acm.org/10.1145/1076034.1076060

[Shneiderman 1996] Shneiderman, Ben: The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In: *IEEE Symposium on Visual Languages* 0 (1996), p. 336. – URL http://doi.ieeecomputersociety.org/10.1109/VL.1996.545307

[Simperl 2009] Simperl, Elena: Reusing ontologies on the Semantic Web: A feasibility study. In: *Data & Knowledge Engineering* 68 (2009), Nr. 10, p. 905–925. – URL http://dx.doi.org/10.1016/j.datak.2009.02.002

[Smeulders et al. 2000] Smeulders, Arnold W. M. ; Worring, Marcel ; Santini, Simone ; Gupta, Amarnath ; Jain, Ramesh: Content-Based Image Retrieval at the End of the Early Years. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22 (2000), Nr. 12, p. 1349–1380. – URL http://dx.doi.org/10.1109/34.895972

[Srivastava and Koehler 2003] Srivastava, Biplav ; Koehler, Jana: Web service composition - current solutions and open problems. In: *ICAPS 2003: International Conference on Automated Planning and Scheduling. Workshop on Planning for Web Services*, URL http://www.isi.edu/info-agents/workshops/icaps2003-p4ws/papers/srivastava-icaps2003-p4ws.pdf, June 2003, p. 28–35

[Treisman 1985] Treisman, Anne: Preattentive processing in vision. In: *Computer Vision, Graphics, and Image Processing* 31 (1985), Nr. 2, p. 156–177. – URL http://dx.doi.org/10.1016/S0734-189X(85)80004-9

[Tzitzikas and Hainaut 2006] Tzitzikas, Yannis ; Hainaut, Jean-Luc: On the Visualization of Large-Sized ontologies. In: *AVI '06: Proceedings of the working conference on Advanced visual interfaces*, ACM New York, NY, USA, 2006, p. 99–102. – URL http://dx.doi.org/10.1145/1133265.1133285

[Uschold et al. 1998] Uschold, Mike ; Healy, Mike ; Williamson, Keith ; Clark, Peter ; Woods, Steven: Ontology Reuse and Application. In: Guarino, Nichola (Editor): *FOIS'98: Proceedings of the 1st International Conference on Formal Ontology in Information Systems*. Amsterdam : IOS press, 1998, p. 179–192. – URL http://www.cs.utexas.edu/users/pclark/papers/fois98.pdf

[Wang et al. 2008] Wang, Xia ; Hauswirth, Manfred ; Vitvar, Tomas ; Zaremba, Maciej: Semantic web services

selection improved by application ontology with multiple concept relations. In: *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing*, ACM New York, NY, USA, 2008, p. 2237–2242. – URL http://doi.acm.org/10.1145/1363686.1364222

[Ware 2004] Ware, Colin: *Information Visualization: Perception for Design*. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 2004. – 438 p. – URL http://portal.acm.org/citation.cfm?id=329554. – ISBN 1-55860-511-8

[Zhang and Norman 1994] Zhang, Jiaje ; Norman, Donald A.: Representations in Distributed Cognitive Tasks. In: *Cognitive Science* 18 (1994), January–March, Nr. 1, p. 87–122. – URL http://dx.doi.org/10.1016/0364-0213(94)90021-3

[Zhou et al. 2007] Zhou, Yu ; Pan, Jian ; Ma, Xiaoxing ; Luo, Bin ; Tao, Xianping ; Lu, Jian: Applying ontology in architecture-based self-management applications. In: *SAC '07: Proceedings of the 2007 ACM symposium on Applied computing*, ACM New York, NY, USA, 2007, p. 97–103. – URL http://doi.acm.org/10.1145/1244002.1244026